Hierarchisches Routingmanagement –
Autonomes Netzwerkmanagement für ein dynamisches Routing unter Berücksichtigung von Qualitätsanforderungen

Dissertation zur Erlangung des
akademischen Grades Doktoringenieur
der Technischen Universität Ilmenau

vorgelegt von

Thomas Volkert

Ilmenau 2015

Datum der Einreichung: 29.06.2015 (vorliegende Revision vom 11.02.2016)
Datum der Verteidigung: 20.01.2016

Gutachter: Prof. Dr.-Ing. habil. Andreas Mitschele-Thiel,
Technische Universität Ilmenau

Univ.-Prof. Dr. rer. nat. Jochen Seitz,
Technische Universität Ilmenau

PD Dr. rer. nat. Oliver P. Waldhorst,
Karlsruher Institut für Technologie

urn:nbn:de:gbv:ilm1-2016000087

Kurzzusammenfassung

Im Fokus dieser Dissertation steht die Frage, wie ein autonom ablaufendes Routingmanagement aussehen kann, um in Netzwerken die Übertragung von Anwendungsdaten unter Berücksichtigung von Qualitätsanforderungen zu ermöglichen. Dabei steht ein dynamisches Routing im Vordergrund, dessen Entscheidungen von der momentanen Verteilung von verfügbaren Linkkapazitäten im Netzwerk abhängen.

Die vorgestellte neuwertige Lösung enthält drei vollständig autonom ablaufende Protokolle. Sie dienen zur Netzwerkunterteilung und automatischen Platzierung von Managementinstanzen, zur Adresszuweisung im Netzwerk sowie zur kontinuierlichen Verteilung von Routingdaten unter den Netzwerkknoten. Dadurch werden alle Routingtabellen aktuell gehalten, sodass sie die momentanen Pfade sowie auch die für jede bekannte Route verfügbaren QoS-spezifischen Eigenschaften beschreiben. Mit Hilfe dieser Daten ist der in dieser Dissertation eingesetzte Routingalgorithmus in der Lage, die Übertragung von Daten von unterschiedlichen Anwendungen unter Beachtung ihrer Qualitätsanforderungen zu ermöglichen. Dabei beeinflusst die aktuell vorliegende Lastsituation im Netzwerk jede notwendige Routingentscheidung.

Trotz der eingeführten komplexen Signalisierungen bleibt das Gesamtsystem kompatibel zu IPv4/v6 und kann somit für die Übertragung von audiovisuellen Daten in heutigen Netzwerken eingesetzt werden. Dabei profitiert die Skalierbarkeit des resultierenden Gesamtsystems von den innerhalb der Signalisierungen des Routingmanagements verwendeten Datenaggregationen.

Der praktische Teil dieser Arbeit ist zweigeteilt. Der erste Teil beschreibt die Software „Homer Conferencing“. Sie ist als eigenständige Lösung für Videokonferenzen und Testumgebung für audiovisuelle Ströme einsetzbar. Mit ihrer Hilfe können qualitative Unterschiede in Übertragungen audiovisuell vorgeführt werden. Die Software bietet zusätzlich grafische Dialoge zur quantitativen Bemessung der Datenströme und Paketverluste. Der zweite praktische Teil beinhaltet die Implementierung des Routingmanagements und setzt die Protokolle auf Paketebene vollständig um. Dies diente als Basis für die durchgeführten quantitativen Evaluierungen. Sie stellen für ausgewählte Basistopologien von IP-Netzwerken den verursachten Signalisierungsaufwand sowie den resultierenden Nutzen beim Einsatz des vorgestellten Routingmanagements dar.

Abstract

This PhD thesis is focused on the question: how can an autonomously working routing management be designed to allow the transmission of application data while considering quality requirements. To answer this the focus is on a dynamic routing, whose decisions depend on the current distribution of available link capacities in the network.

The presented new solution contains three protocols which work in a completely autonomous way. They are used to cluster the network and place automatically management instances, to assign addresses in the network as well as to distribute continuously routing data among the network nodes. Based on this, all routing tables are kept up to date, so that they represent the current paths as well as they also describe the available QoS specific capacity for each known route. By the help of this data, the routing algorithm, which is applied in this PhD thesis, allows the transmission of data from different applications while considering their quality requirements. In this context, each needed routing decision is influenced by the currently existing load situation in the network.

Despite the introduced complex signaling, the overall system remains compatible to IPv4/v6. Therefore, it can be used for the transmission of audiovisual data in today's networks. In such a scenario the scalability of the resulting overall system is supported by the data aggregations which are used within the signaling of the routing management.

The practical part of the work is divided into two areas. The first one describes the software "Homer Conferencing". It is usable as standalone solution for video conferences and test environment for audiovisual streams. By its help, qualitative differences in transmissions can be presented. Additionally, the software provides graphical dialogs for quantitative measurements of the data streams and packet losses. The second practical part contains the implementation of the routing management and applies all protocols on packet level. This was used as base for the accomplished quantitative evaluations. They show the caused signaling overhead as well as the resulting benefit of the introduced routing management for selected base topologies of IP networks.

„Die Zeit verweilt lange genug für denjenigen, der sie nutzen will.“

— Leonardo da Vinci

Danksagung

Nach dem im Jahr 2009 die ersten Ideen entstanden, ist es nun soweit, nach etwa 6 Jahren schreibe ich nun diese letzten Zeilen – ein spannender Moment. Ich hoffe, für jeden Leser ist diese Arbeit mindestens genauso spannend wie für mich die Zeit des Konzipierens und sogar des Schreibens war. Rückblickend betrachtet sehen viele Dinge einfacher aus als sie es tatsächlich waren. Ich möchte diese Stelle für ein paar persönliche Worte verwenden und mich bei den zahlreichen Menschen bedanken, die diese Arbeit bzw. mich unterstützt haben. Anfangen möchte ich bei allen Kollegen/Freunden aus dem Fachgebiet „Integrierte Kommunikationssysteme“ der TU Ilmenau – es war eine wirklich spannende Zeit mit vielfältigen neuen Erfahrungen für mich. Besonderer Dank gilt dabei Prof. Dr.-Ing. habil. Andreas Mitschele-Thiel, der meinem Kollegen Florian Liers und mir die Freiheit gab, zusammen den Forschungsansatz „Forwarding on Gates“ zu starten und als gefördertes Projekt tatsächlich umzusetzen. Des Weiteren gab er mir die Möglichkeit, das Konzept des in dieser Dissertation vorgestellten Routingmanagements zu entwickeln. Es brauchte einige Zeit, um alle Herausforderungen für ein autonomes System zu bewältigen. Jetzt kann ich aber mit ruhigem Gewissen sagen: Ja, es geht. Bei der Entwicklung half sicherlich auch die Weiterentwicklung der Videokonferenzsoftware „Homer-Conferencing“ – vielen Dank an der Stelle für die Möglichkeit zur Fortsetzung. Die Software brachte zusätzliche Motivation und führte auch zu dem Leitspruch: Das muss gehen!

Zusätzlich möchte ich Univ.-Prof. Dr. rer. nat. Jochen Seitz und PD Dr. Oliver Waldhorst für ihr Interesse an der vorliegenden Dissertation danken und hoffe, die Arbeit ist trotz der etwas längeren Fassung dennoch spannend zu lesen.

Meinem ehemaligen Kollegen und Büronachbarn Florian Liers möchte ich für die zahlreichen inhaltlich wertvollen Diskussionen danken – die Bürotafeln waren immer ein gern genutztes Werkzeug. Ebenso möchte ich Jürgen Schmidt danken, der nicht nur für eine funktionierende Infrastruktur sorgte.

Ebenso möchte ich allen (ehemaligen) Studenten danken, die entweder direkt oder indirekt zu dieser Arbeit beigetragen haben. Eine vollständige Liste aller Arbeiten befindet sich am Ende dieser Dissertation. Ein spezieller Dank gilt dabei Stefan-Wieland Kögel und Manuel Osdoba, die mit ihrer Diplom- bzw. Masterarbeit einen besonderen Beitrag für diese Arbeit geleistet haben.

Des Weiteren möchte ich meinen Freunden Denny Duphorn, Alexander Krause, Tino Schmidt sowie Tim Langner für ihren unermüdlichen Einsatz beim Kommentieren meiner Texte danken.

Besonderen Dank gilt auch Marcel Pennewiß, der nicht nur durch seine Bachelorarbeiten beigetragen hat, sondern auch bei der Fehlerbereinigung bei Homer-Conferencing half. Des Weiteren brachte er die Software zu Gentoo Linux.

Zusätzlich geht auch ein Dankeschön an die Blender Foundation als Urheber des „Big Buck Bunny“-Videos und die Erlaubnis zur kostenlosen Nutzung für Demonstrationszwecke (Screenshots sind auch in dieser Arbeit enthalten).

Auch meinen Auftraggebern der letzten 2 Jahre gilt mein Dank, sie ermöglichten über ihre Projekte indirekt die Fertigstellung dieser Arbeit.

Weiterhin möchte ich meiner Familie für ihre unermüdliche Geduld und das Verständnis in unzähligen Momenten meiner Abwesenheit danken – viele Dinge mussten wegen der Dissertation liegen bleiben und ich war oftmals zu beschäftigt. Schlussendlich möchte ich meiner Lebensgefährtin Katja für ihre Geduld und das Verständnis danken – besonders für die letzten Monate. Jetzt wird mehr Zeit für alles sein.

Inhaltsverzeichnis

1 Einleitung

Im Laufe der Entwicklung des Internets entstanden stetig neue Anwendungsgebiete. Es entwickelte sich immer mehr zum Massenkommunikationsmedium. Das Angebot umfasste nach und nach auch große Datenbanken für Informationsrecherchen, individuelle Spezialsoftware (Kauf- und Downloadmöglichkeiten), Portale zum Handel mit Produkten und Dienstleistungen (E-Commerce), sowie Onlinediskussionen. Besonders rasch verbreiteten sich Multimediaangebote. Sie beeinflussen bis heute signifikant das Verhalten der Nutzer als auch deren Anforderungen an das Internet. Zu diesen Angeboten zählen Dienste, welche zum einen die Bereitstellung aufgezeichneter Videofilme oder Musikstücke und zum anderen das Ansehen von Liveübertragungen ermöglichen. Aufgrund ihrer steigenden Beliebtheit nahm insbesondere in den letzten Jahren der Anteil an Multimediadaten im Internet rasant zu. Im Jahr 2011 bestand bereits 51% des im Internet durch Privatanwender verursachten Datenverkehrs (exklusive *Peer-to-Peer* Austausch von Videomaterial) aus Videodaten [1]. Prognosen sagen voraus, dass dieser Anteil bis Ende 2018 auf etwa 79% ansteigen wird. Betrachtet man hingegen alle Formen von Videoübertragungen (inklusive *Peer-to-Peer*-Übertragungen), wird der Anteil am Gesamtdatenaufkommen, verursacht durch Privatanwender, laut Prognosen sogar bis zu 90% im Jahr 2018 einnehmen [2]. Ein maßgebender Beitrag ist dabei durch die Angebote von *Netflix* zu erwarten, dies kann beispielsweise an aktuellen Untersuchungen [3] für Nordamerika abgelesen werden. Durch dessen Einführung am 16. September 2014 in Deutschland, Frankreich, Österreich, Schweiz, Belgien und Luxemburg ist eine ähnliche Entwicklung für europäische Netzwerken zu erwarten – das Datenaufkommen durch Multimediadaten wird auch in Zukunft signifikant steigen.

Bei der Übertragung von Multimediadaten ist jedoch nicht nur eine zuverlässige Bereitstellung des Dienstes wichtig. Vielmehr zählt für den Nutzer auch die Qualität der lokalen Wiedergabe des übertragenen Inhaltes: Ein Video sollte möglichst ohne Aussetzer und mit klar verständlichem Ton abgespielt werden. Aus technischer Sicht ist dafür eine Übertragung mit einer stabilen und ausreichend hohen Datenrate wichtig. Während man beim Konsumieren von serverseitig gespeicherten Multimediadaten auf die Ankunft weiterer Inhalte warten kann, ist dies bei Live-Übertragungen kritischer zu sehen. Insbesondere während Videokonferenzen möchte jeder Teilnehmer eine gute Interaktivität erfahren. Dafür ist es notwendig, dass die Multimediadaten mit ausreichender Qualität innerhalb einer vorgegebenen Zeit beim Empfänger eintreffen. Technisch werden dafür Netzwerkübertragungen mit entsprechend definierten Eigenschaften benötigt, sodass eine als akzeptabel empfundene Qualität des Dienstes beim Nutzer erreicht wird.

1.1 Motivation

In heutigen Netzwerken werden Konzepte eingesetzt, welche bereits im ursprünglichen ARPANET entwickelt worden sind. Prominentes Beispiel dafür ist das Internet Protokoll (IP), welches für die Übertragung von Anwendungsdaten auf der Basis von Paketen eingesetzt wird. Dabei werden Entscheidungen über den zu nutzenden Netzwerkpfad durch Zwischenknoten getroffen. Erst die Kombination ihrer Einzelentscheidungen legt den resultierenden Pfad, die Route, von der Quelle zum Ziel fest. Als Basis dieser Routingentscheidungen dienen bekannte Kostenwerte, welche den Aufwand bei Nutzung der jeweiligen Route widerspiegeln. Das für IP-basierte Netzwerke typische Modell verwendet die Anzahl der entlang einer Route zu passierenden Knoten als primären Kostenfaktor. Die Anwendung dieser Kostenzuordnung führt zu dem sogenannten *Best Effort Routing* (BE-Routing), welches Pakete auf möglichst kurzem Weg zum Ziel leitet.

Wie bereits zu Beginn beschrieben, stellt die Übertragung von Multimediadaten an Netzwerke, insbesondere das Internet, zusätzliche Anforderungen an die Eigenschaften der Übertragung. Zu diesem Zweck wurden zwei grundsätzliche Modelle für die Erbringung von Dienstqualität entwickelt: *Differentiated Service* (DiffServ) [4] und *Integrated Service* (IntServ) [5]. Ersteres eignet sich zur einfachen

Priorisierung von Datenströmen – jedoch ermöglicht es keine zuverlässigen Qualitätszusagen. Im Gegensatz dazu arbeitet *IntServ* auf der Basis von festen Ressourcenreservierungen pro Strom, wodurch notwendige Linkkapazitäten einem Datenstrom fest zugeordnet werden und somit zuverlässige Qualitätszusagen ermöglicht werden. Des Weiteren sind hybride Systeme möglich, welche die Vorteile beider Varianten kombinieren. Unabhängig vom eingesetzten Modell ist die resultierende Pfadauswahl, das Routing, von der Quelle zum Ziel für die resultierende Qualität der Übertragung entscheidend. Ein ausgeklügeltes Routing reagiert dynamisch auf Anforderungen einer Anwendung und die veränderten Linkkapazitäten im Netzwerk. Es leitet Daten einer Anwendung entlang einer Route, welche die gewünschte Dienstqualität ermöglicht.

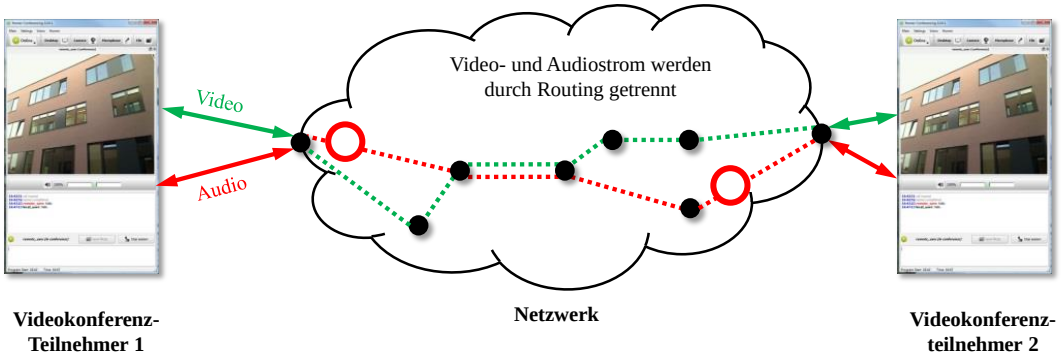

Abbildung 1.1: Routing in Abhängigkeit von Anforderungen der Anwendung und Kapazitäten im Netzwerk

Wie in Abbildung 1.1 dargestellt, muss keinesfalls stets die kürzeste Route mit der geringsten Anzahl von Zwischenknoten gewählt werden. Lokale Engpässe können dazu führen, dass die Wahl einer kürzeren Route – im Vergleich zu einer alternativen längeren Route – zu einer geringeren Datenrate von erfolgreich übertragenen Paketen führt. Beispiele dafür sind in Abbildung 1.1 durch rote Kreise markiert. Der Audiostrom belastet im dargestellten Szenario die rote Route zusätzlich, sodass die verbleibenden Linkkapazitäten an den markierten Stellen eine zu geringe Datenrate für die zusätzliche Übertragung der Videodaten zulassen. Das kann zu Bildfehlern und -ausfällen auf der Empfängerseite führen. Um dies zu vermeiden sollte der Videostrom entlang der grün markierten Route durch das Netzwerk geleitet werden, welche trotz der höheren Knotenanzahl die besseren Eigenschaften bietet.

1.2 Zielstellung

Die vorliegende Arbeit beantwortet die Frage, wie ein autonomes Routingmanagement aussehen kann, um in Netzwerken die Übertragung von Anwendungsdaten unter Berücksichtigung von Qualitätsanforderungen zu ermöglichen. Dabei soll ein dynamisches Routing zum Einsatz kommen, dessen Entscheidungen von der momentanen Verteilung von frei verfügbaren Linkkapazitäten im Netzwerk abhängen. Im Fokus steht eine automatisierte Lösung:

- zur Netzwerkstrukturierung und Platzierung von Managementinstanzen,
- zur Adresszuweisung im Netzwerk,
- für die kontinuierliche Verteilung von Routingdaten sowie
- zum Routing von Anwendungsdaten unter Beachtung von Qualitätsanforderungen.

Aus Nutzersicht steht eine dynamische Pfadauswahl im Netzwerk im Vordergrund, welche den Anforderungen der jeweiligen Anwendungen an Datenrate und Verzögerung im Rahmen der verfügbaren Kapazitäten im Netzwerk gerecht wird. Dadurch erhält eine Anwendung Qualitätszusicherungen für ihre Übertragungen. Insbesondere im Kontext der Übertragung von audiovisuellen Daten kann dadurch eine möglichst gute Präsentationsqualität auf Empfängerseite unterstützt werden.

Aus Sicht eines Netzwerkbetreibers soll das Routingmanagement einen neuartigen Schritt in Richtung eines autonom arbeitenden Netzwerkmanagements repräsentieren. Die verwendeten Signalisierungen dürfen dabei keine manuellen Eingaben durch einen menschlichen Netzwerkadministrator benötigen, dennoch sollen manuelle Eingriffe zur Umsetzung von festgelegten Netzwerkrichtlinien unterstützt werden. Das resultierende Gesamtsystem soll dabei jedoch kompatibel zu heutigen IP-basierten Netzwerken verbleiben.

1.3 Eingrenzung der Zielstellung

Für das Routing von Paketen einer Anwendung steht im Kontext von OSPF-basierten Netzwerken und seinen QoS-spezifischen Erweiterungen die beste Lösung in Abhängigkeit von den jeweiligen Zielstellungen im Vordergrund. Dafür müssen detaillierte Topologiedaten auf jedem Router des jeweiligen Netzwerks gespeichert werden [6] [7] [8]. Die vorliegende Arbeit unterscheidet sich von dieser Herangehensweise insofern, dass nicht nur die kürzeste Route, oder alternativ, jene mit der größten noch verfügbaren Datenrate gesucht wird. Stattdessen wird ein Kompromiss aus Signalisierungsaufwand zur Verteilung von Routingdaten und Effizienz der Verteilung von Datenströmen im Netzwerk verwendet.

Die Bestimmung der Eigenschaften von physikalischen Links steht nicht im Fokus. Die Ausführungen beinhalten die Annahme, dass ein Verfahren existiert, mit dessen Hilfe man die Kapazitäten von Links bestimmen kann.

Der Fokus liegt ebenfalls nicht auf einer optimalen Verteilung von Lasten für ein Netzwerk. Stattdessen werden globale Statusinformationen vermieden und Berechnungen auf der Basis von begrenzten Routingdaten mit reduziertem Detailgrad durchgeführt. Dadurch wird ein ausgewogener Kompromiss zwischen Komplexität der Signalisierungen und der daraus resultierenden Güte des Routings erreicht, sodass ausgelastete Routen vermieden und durch bekannte Alternativen ersetzt werden.

Des Weiteren wird durch die vorliegende Arbeit kein vollständig neuer Routingalgorithmus eingeführt. Stattdessen werden bekannte Ansätze kombiniert, um möglichst alle verfügbaren Kapazitäten des Netzwerks für eine Routingentscheidung einzubeziehen. Dabei werden ungeeignete oder überlastete Routen vermieden, um den Qualitätsanforderungen der Anwendung gerecht zu werden. Der Algorithmus soll sowohl für das *IntServ*- als auch das *DiffServ*-Modell einsetzbar sein, wobei der Fokus im Kontext dieser Arbeit auf dem *IntServ*-Modell und den damit verbundenen festen Ressourcenreservierungen je aufgebauter Verbindung liegt. Des Weiteren soll das resultierende Routingsystem sowohl für IP-basierte Netzwerke als auch mögliche Alternativen einsetzbar sein.

Ebenfalls beinhaltet die vorliegende Arbeit keine Alternative zu heutigen Protokollen zur Ressourcenreservierung. Das in dieser Arbeit vorgestellte Konzept ist stattdessen allgemeingültig und kann eine geeignete Route unter Beachtung der geforderten Ressourcen bestimmen.

1.4 Wissenschaftliche Leistungen

Das Kernthema dieser Arbeit ist das Konzept eines neuartigen Routingmanagements, welches sich von anderen Lösungen insbesondere durch seine autonom ablaufenden Prozesse zum Netzwerkmanagement unterscheidet. Es beinhaltet sowohl eine Verteilung von Adressen als auch von Routingdaten im Netzwerk. Darauf aufbauend bietet es ein Routing, dessen Entscheidungen die Anforderungen der jeweiligen Anwendung einbeziehen, sodass die Anwendungsdaten mit der gewünschten Qualität für die resultierende Datenrate und Gesamtverzögerung übertragen werden. Dabei wird jede Routingentscheidung auf aktuellen Daten über vorhandene Routen und ihren jeweils noch zur Verfügung stehenden QoS-spezifischen Eigenschaften getroffen.

Während das Routingmanagement universell einsetzbar ist, steht im Kontext dieser Arbeit seine Anwendung für das *IntServ*-Modell und den damit verbundenen festen Ressourcenreservierungen je Datenstrom einer Anwendung im Vordergrund. Trotz seines Funktionsumfanges und Komplexität ist das Gesamtsystem dabei auch für große Netzwerke skalierbar, da sowohl der Speicheraufwand als auch die Berechnungslast im Netzwerk zu jeder Zeit verteilt werden. Des Weiteren verbleibt das System insgesamt kompatibel zu IPv4/IPv6 und kann somit in heutigen IP-basierten Netzwerken problemlos eingesetzt werden.

Zu den Kernkomponenten des in dieser Arbeit vorgestellten innovativen Konzeptes für ein neues Routingmanagement gehören:

- **Autonome Verteilung von Netzwerkmanagementdaten:** Mit Hilfe von drei unterschiedlichen Protokollen werden durch HRM benötigte Daten zur Netzwerkverwaltung und dem darauf aufbauenden Routing signalisiert. Jede dieser Signalisierungen ist gegen Paketverluste abgesichert und der Empfang von Nachrichten in korrekter Reihenfolge ist für alle Übertragungen sichergestellt, sodass dadurch eine zuverlässige Synchronisation der Abläufe auf verschiedenen Knoten gewährleistet wird. Dabei werden für die Übertragung von einzelnen Signalisierungen ausschließlich die verteilten Daten des Managementsystems sowie des zugrundeliegenden Protokolls von Schicht 2 (bezogen auf das OSI-Modell) verwendet. Diese Eigenschaft unterscheidet das System grundsätzlich von bisherigen Lösungen, sodass im Gegensatz zu den Signalisierungen von OSPF oder BGP weder vorgegebene Adressen von Schicht 3 (IP-Adressen) noch eine manuell festgelegte Netzwerkstrukturierung (Unterteilung in Subnetze) benötigt werden. Zu den Protokollen gehören:
    1. **Protokoll zur Platzierung von Managementinstanzen:** Das Routingmanagement basiert auf einem Overlay-Netzwerk aus Managementinstanzen, welche im Netzwerk platziert werden müssen. Dies erfolgt autonom mit Hilfe eines eigens dafür entwickelten Signalisierungsprotokolls. Im Gegensatz zu bestehenden Ansätzen benötigt es weder eine vorkonfigurierte Adressverteilung noch eine manuelle Netzwerkunterteilung. Des Weiteren beachtet es die vorhandene Netzwerktopologie und platziert die Managementinstanzen unter Berücksichtigung der resultierenden Kommunikationswege, wodurch diese möglichst kurz gehalten werden. Zusätzlich reagiert das Protokoll im Fall von Topologieänderungen oder Ausfällen automatisch, sodass die Verteilung der Instanzen automatisch angepasst und stets das gesamte Netzwerk durch das Management erfasst wird.
    2. **Protokoll zur Adressvergabe:** Zur Festlegung des Ziels von Anwendungsdaten sowie für Routenberechnungen müssen den Knoten im Netzwerk eindeutige Adressen zugeordnet werden, sodass sie eindeutig als Ziel identifizierbar sind. Zu diesem Zweck werden mit Hilfe eines eigens dafür entwickelten zweiten Signalisierungsprotokolls jedem Knoten eine oder mehrere eindeutige Adressen automatisch zugewiesen. Das Protokoll verwendet dabei die zuvor platzierten Managementinstanzen und die dadurch vorgegebene Strukturierung des Netzwerks zur hierarchischen Verteilung von Adressen. Im Gegensatz zu bisherigen Systemen zur Adressvergabe erhält dabei jeder Router seine Adresse(n) ohne manuelle Vorgaben eines Netzwerkadministrators. Die verwendete Adressierung bleibt zudem kompatibel zu bekannten IPv4/v6-Adressen und kann in diese Adressierungsschemata integriert werden, wodurch ein Einsatz in heutigen Netzwerken unterstützt wird und ebenfalls eine direkte Interoperation möglich ist.
    3. **Protokoll zur Verteilung von Routingdaten:** Mit Hilfe des dritten entwickelten Signalisierungsprotokolls werden zwischen den Managementinstanzen ausgewählte Routingdaten verteilt. Sie beschreiben existierende Pfade durch das Netzwerk und beinhal-

ten zusätzlich Informationen über die jeweils aktuell verfügbaren QoS-spezifischen Eigenschaften einer Route. Dazu zählen die noch verfügbare Datenrate und die zu erwartende Verzögerung. Diese Daten werden auf den Knoten als lokale Routingtabelle abgespeichert, welche wiederum als Eingabe für die Routingentscheidungen des jeweiligen Knotens dienen.
Innerhalb der Signalisierungen werden verschiedene Methoden zur Aggregation von Routen verwendet. Dadurch erreicht die Kontrollebene eine zusätzliche Reduktion des verursachten Signalisierungsaufkommens im Netzwerk zum Preis von reduzierten Topologiedetails innerhalb der Routingtabellen.

- **Routing von Anwendungsdaten:** Der in dieser Arbeit entwickelte Routingalgorithmus stellt das Herzstück für das Routing von Anwendungsdaten dar. Er bietet insbesondere folgende zwei Vorteile:
    1. **Beachtung von Qualitätsanforderungen beim Routing:** Jede Routingentscheidung wird unter Beachtung der von der Anwendung geforderten Datenrate und der maximal erlaubten Gesamtverzögerung getroffen. Dabei werden ungenügende Pfade durch das Netzwerk vermieden, insofern noch eine Route zum Ziel mit ausreichenden Ressourcen bekannt ist.
    2. **Beachtung von aktuellen QoS-spezifischen Routeneigenschaften beim Routing:** Für Routingentscheidungen werden neben der Routenlänge ebenso die aktuell verfügbare Datenrate sowie die zu erwartende Verzögerung von jeder Route einbezogen. Diese Daten werden durch die Managementinstanzen zwischen den Knoten des Netzwerks signalisiert. Dadurch hängt jede Routingentscheidung von der aktuellen Kapazitätsverteilung im Netzwerk ab.

  Auf Basis dieser Eingaben vermeidet der Routingalgorithmus frühzeitig ungeeignete Pfade und verhindert Engpässe im Netzwerk. Dadurch wird die geforderte Dienstqualität für Anwendungen sichergestellt. Diese Eigenschaft unterscheidet das Routingmanagement von dem in IP-basierten Netzwerken typischen *Best Effort Routing*. Des Weiteren beachtet das Routingmanagement bei der Wegewahl die Fairness zwischen Übertragungen, sodass ein Datenstrom möglichst wenige parallele Datenströme und die damit verbundenen Routingentscheidungen beeinflusst oder gar blockiert. Zu diesem Zweck werden Routen möglichst kurz gewählt. Nur bei Ressourcenknappheit wird auf längere Wege ausgewichen, welche größere Teile des Gesamtnetzwerks durchqueren. Alle vorhandenen Netzwerkressourcen werden somit für einzelne Anwendungen nutzbar. Durch diese Eigenschaft unterscheidet sich das Routingmanagement dieser Arbeit von typischen IP-basierten Lösungen grundsätzlich. Sie favorisieren stattdessen den kürzesten Weg oder unterstützen nur einfache Lastverteilungsstrategien, welche Pakete abwechselnd entlang verschiedener Routen mit gleichen Kosten zum Ziel leiten.

Trotz seiner komplexen Signalisierungsvorgänge verbleibt das Routingmanagement als eigenständige Lösung, welche unabhängig von vorhandenen Protokollen von Schicht 3 arbeitet. Folglich werden weder IP-Adressen noch vorgegebene Unterteilungen des Netzwerks benötigt. Dabei verbleibt das Gesamtsystem dennoch kompatibel zu IPv4 und IPv6. Das im vorgestellten Routingmanagement angewandte Adressierungsschema kann zudem in das IP-spezifische Schema integriert werden, sodass das Gesamtsystem ohne Einschränkungen in heutigen Netzwerken zum Einsatz kommen kann. Des Weiteren wird durch die vorliegende Arbeit auch gezeigt, dass es sich ebenfalls für FoG-basierte Netzwerke [9] als sinnvolle Erweiterung in Form eines eigenständigen Routingdienstes eignet. Bei der Konzeption spielte Skalierbarkeit eine wichtige Rolle. Das Routingmanagement ist neben dem Einsatz in Firmennetzwerken auch für eine Anwendung in Providernetzwerken gerüstet. Dabei spielt insbesondere die zur Verwaltung der Managementinstanzen eingesetzte Hierarchie eine vordergründige Rolle. Sie wird für die

hierarchische Verteilung von Adressen als auch von notwendigen Routingdaten verwendet. In Kombination mit den eingesetzten Aggregationsmechanismen wird dadurch eine für die gewünschten Anwendungsfälle gute Skalierbarkeit erzielt.

Neben den detailreichen konzeptionellen Beschreibungen beinhaltet die vorliegende Arbeit ebenfalls umfangreiche Implementierungsleistungen. Dazu zählen:

- **Erweiterungen für den Netzwerksimulator *FoGSiEm*** [10]: Der erste praktische Teil enthält eine Implementierung des konzipierten Routingmanagements. Für eine einfache Beobachtung und Nachvollziehbarkeit der durch die Managementinstanzen durchgeführten Signalisierungen stehen vielfältige grafische Ausgabedialogen zur Verfügung, welche ebenfalls den aktuellen Status des Managementsystems in Echtzeit visualisieren. Des Weiteren bieten die eigens dafür integrierten Softwaresensoren die Möglichkeit zur automatischen Ermittlung von Messwerten für alle drei verwendeten Signalisierungsprotokolle. Diese Funktionalität wird innerhalb des Evaluierungsteils dieser Arbeit für Messungen anhand ausgewählter Topologien verwendet. Auf Basis dieser Ergebnisse sind weiterführende Vergleiche mit alternativen Routingsystemen zukünftig möglich.
- **Programmierschnittstelle für Anwendungen:** Zur Nutzung des entworfenen Routingmanagements in zukünftiger Anwendungssoftware wird eine Programmierschnittstelle vorgestellt und innerhalb der Implementierung erprobt, die es einer Anwendung ermöglicht, für eine Übertragung Qualitätsanforderungen festzulegen und verbleibende Kapazitäten des Netzwerks abzufragen.
- **Multimediatestbett *Homer-Conferencing*** [11]**:** Im letzten Implementierungsteil wird eine Software für Videokonferenzen präsentiert, welche während der vorgestellten Forschungsarbeit im Rahmen dieser Arbeit entstand. Sie steht als eigenständige Testplattform für audiovisuelle Übertragungen und zum Studium sowie für Messungen von Datenströmen am konkreten Anwendungsbeispiel zur Verfügung. Dabei unterscheidet sich *Homer-Conferencing* von alternativer Software insbesondere durch die flexible Anbindung von Netzwerkprotokollen, wodurch ein Vergleich von verschiedenen Netzwerkstacks sowie Routingvarianten ermöglicht wird. Die Software bietet zudem verschiedene grafische Dialoge zur expliziten Festlegung von Qualitätsanforderungen für die Übertragung von audiovisuellen Daten, welche durch das Routing im Netzwerk beachtet werden. Als Gegenstück dazu können auf der Empfängerseite die charakteristischen Werte der eintreffenden Datenströme mit Hilfe von grafischen Dialogen der Software in Echtzeit beobachtet werden und ebenfalls automatisch Statistiken erzeugt werden. Des Weiteren stellt die Software die notwendigen Funktionen zur Messung des Datenaufwands für die Signalisierung von Qualitätsanforderungen zwischen der Anwendung und den Routern des Netzwerks bereit. Die Funktionen von Homer-Conferencing wurden im Rahmen dieser Arbeit insbesondere zum Studium der qualitativen Unterschiede zwischen Übertragungen entlang verschiedener Routen eingesetzt. Sie wurde ebenfalls mehrfach in öffentlichen Vorführungen verwendet.

1.5 Struktur dieser Arbeit

Im nachfolgenden Kapitel 2 wird der Stand der Technik zu bekannten Konzepten für die paketbasierte Übertragung von Anwendungsdaten erläutert. Zu Anfang werden wichtige Aspekte sowie praxisrelevante Lösungen heutiger Netzwerke vorgestellt, anschließend werden darauf aufbauend bekannte Erweiterungen für ein Routing von Anwendungsdaten unter Beachtung von Qualitätsanforderungen beschrieben. Aus diesem Kontext heraus werden in Abschnitt 2.2.3 Anforderungen an ein Routing von

audiovisuellen Datenströmen aufgestellt, welche im weiteren Verlauf dieser Arbeit berücksichtigt werden. Das Kapitel schließt inhaltlich mit der Beschreibung ausgewählter Forschungsarbeiten für zukünftige Netzwerke.

Kapitel 3 stellt den Theorieteil dieser Arbeit dar, worin das Konzept eines neuen Routingmanagements beschreiben wird. Auf Basis der vorgestellten Signalisierungen werden ein autonomes Netzwerkmanagement sowie ein dynamisches Routing ermöglicht, sodass dadurch Anwendungsdaten unter Beachtung von Qualitätsanforderungen durch das Netzwerk geroutet werden können. Das Kapitel beginnt dabei mit einer Aufstellung der Anforderungen an das Managementsystem. Anschließend wird ein Überblick über die neue Gesamtarchitektur gegeben. Die darin beinhalteten zentralen Prozesse des Systems werden entsprechend ihrer kausalen Abhängigkeit nacheinander vorgestellt. Dabei kommt insbesondere die autonome Arbeitsweise des Gesamtsystems zum Ausdruck. Im Anschluss wird in Abschnitt 3.9 eine mögliche Integration in heutige IP-basierte Netzwerke beschrieben. Abschnitt 3.10 enthält eine Diskussion des Gesamtkonzeptes, dabei wird insbesondere in Abschnitt 3.10.9 das Gesamtkonzept mit den eingangs vorgestellten Anforderungen verglichen und die Frage beantwortet, wie diese durch die Architektur berücksichtigt werden. Zum Ende des Kapitels erfolgt in Abschnitt 3.10.9 ein umfassender Vergleich mit bisher bekannten Ansätzen.

Der Praxisteil dieser Arbeit ist in zwei größere Bereiche unterteilt und erstreckt sich über die Kapitel 4 und 5. Im ersten Teilt wird ein Überblick über die Implementierung des Routingmanagements gegeben. Das Kapitel beginnt mit einer Aufstellung der Anforderungen an die Implementierung. Im Anschluss wird die implementierte Softwarearchitektur anhand ausgewählter Bereiche vorgestellt. Dabei wird die Unabhängigkeit des Konzeptes von den Eigenheiten heutiger als auch alternativer zukünftiger Netzwerkkonzepte verdeutlicht. Die Erläuterungen enthalten zudem implementierungsspezifische Erweiterungen des Konzeptes, welche insbesondere die Konvergenzzeit des Gesamtsystems bei Topologieänderungen zusätzlich verkürzen. Ab Abschnitt 4.4 liegt der Fokus auf der Anwendung des Routingmanagements. Dabei wird zuerst die integrierte Programmierschnittstelle vorgestellt, welche die Funktionen des umgesetzten Konzeptes gegenüber Anwendungssoftware zugänglich machen. Anschließend erfolgt die Vorstellung von zusätzlich integrierten Testanwendungen, welche die vorgestellte Programmierschnittstelle verwenden. Das Kapitel schließt inhaltlich in Abschnitt 4.6 mit einem Vergleich zwischen den ursprünglich aufgestellten Anforderungen und der verfügbaren Implementierung.

In Kapitel 5 wird der zweite Praxisteil dieser Arbeit beschrieben. Er konzentriert sich auf ein Multimediatestbett für audiovisuelle Übertragungen. Die darin vorgestellte Anwendung existiert heute als eigenständige Open-Source-Videokonferenzlösung und diente während der Konzept- und Implementierungsphase als Instrument für Studien bzw. Präsentationen. Ähnlich Kapitel 3 und 4 beginnt die Vorstellung der Anwendung mit einer Aufstellung der Anforderungen, um einen Überblick über die Zielstellung zu geben. Im Anschluss werden die Softwarearchitektur sowie die einzelnen Module der Anwendung erläutert. Abschnitt 5.7 stellt insbesondere die partielle Integration in die Software aus Kapitel 4 vor, sodass eine Lösung für Videostreaming innerhalb der Netzwerksimulation verfügbar ist. Das Kapitel schließt inhaltlich in Abschnitt 5.8 mit einer Diskussion der zuvor aufgestellten Anforderungen und beschreibt, wie diese innerhalb der Implementierung berücksichtigt werden.

Kapitel 6 beschreibt die durchgeführten Netzwerksimulationen zur Evaluierung des Routingmanagements. Die dabei durchgeführten Messungen werden anhand von Grafiken vorgestellt und die Kurvenverläufe begründet. Dabei werden sowohl Kosten als auch Nutzen des Gesamtsystems näher beleuchtet. Zusätzlich wird die Anwendung von Netzwerkrichtlinien erläutert sowie die resultierende Genauigkeit und Aktualität von Routingdaten auf Knoten diskutiert. Das Kapitel schließt mit einem Vollständigkeits- und Konsistenztest, der das Routingmanagement zusätzlich für ein reales Netzwerk anwendet.

Abschließend werden die Ergebnisse dieser Arbeit zusammengefasst und mögliche zukünftige Themenstellungen vorgestellt, die zur Fortsetzung dieser Arbeit dienen können.

1.6 Leseeinstieg und Nachvollziehbarkeit

Innerhalb der textuellen Beschreibungen von Kapitel 2 werden wichtige Begriffe der Arbeit in Zusammenhang gebracht und ausgewählte Mechanismen für aktuelle sowie mögliche zukünftige Netzwerke im Überblick erläutert. Der technisch bewanderte Leser kann den Einstieg in Abschnitt 2.2.3 ansetzen. Hier werden die Qualitätsanforderungen an Übertragungen diskutiert, welche während des Routings beachtet werden müssen. Auf diese wird im weiteren Verlauf der Arbeit verwiesen.

Die Ausführungen sind oftmals bebildert, um neben textueller Beschreibung auch durch visuelle Darstellungen das Gesamtverständnis des Lesers zu unterstützen. Insbesondere werden in Kapitel 3 häufig Grafiken zur Veranschaulichung eingesetzt. Darin werden die Abläufe anhand eines konkreten Praxisbeispiels dargestellt. Das verwendete Szenario verbleibt über das gesamte Kapitel 3 gleich, um eine bessere Nachvollziehbarkeit notwendiger Einzelschritte zum Aufbau des Gesamtsystems zu ermöglichen.

Innerhalb der Kapitel 3 bis 5 wurde eine ähnliche Struktur gewählt. Darin sind grundsätzlich folgende Kerninhalte wiederzufinden: Anforderungen an die Architektur, Architekturbeschreibung, detailliertere Beschreibung einzelner Komponenten sowie eine Bewertung der umgesetzten Anforderungen.

Innerhalb der Beschreibungen der durchgeführten Experimente von Kapitel 5 wurde darauf geachtet, dass sowohl die Grenzen, die zu akzeptierenden Kosten als auch der Nutzen des Gesamtsystems beleuchtet werden. Des Weiteren erschien es wichtig, zum Ende der Arbeit ein Beispiel für reale Netzwerke vorzustellen, um den praktischen Nutzen dieser Arbeit abschließend zu betonen.

2 Stand der Technik – Datenübertragung in Netzwerken

Dieses Kapitel vermittelt ein grundsätzliches Verständnis der allgemeinen Datenübertragung in heutigen Netzwerken und stellt zusätzlich ausgewählte Ansätze für zukünftige Netzwerke vor. Es werden benötigte Begriffe eingeführt, welche im weiteren Verlauf dieser Arbeit zur Anwendung kommen. Die Ausführungen erheben nicht den Anspruch auf Vollständigkeit, stattdessen werden die für das Verständnis dieser Arbeit wichtigsten Aspekte notwendiger Abläufe und Signalisierungen erläutert. Im Vordergrund der Ausführungen steht insbesondere das Routing von zu übertragenden Daten unter Berücksichtigung von Qualitätsanforderungen der Anwendung.

2.1 Heutige Netzwerke

Zur Übertragung von Anwendungsdaten in heutigen Netzwerken werden diese in Fragmente unterteilt, welche jeweils innerhalb eines Pakets durch das Netzwerk geleitet werden.

2.1.1 Protokolle, Weiterleitung und Routing

Ein Paket enthält neben den Anwendungsdaten auch sogenannte Metadaten. Typischerweise werden diese zu Beginn im vorderen Teil, dem Paketkopf (Englisch: „packet header“), übertragen. Auf Basis dieser Daten werden festgelegte Abläufe im Netzwerk durchgeführt, welche die Zustellung am gewünschten Empfänger sicherstellen. Die Definition dieser Prozesse sowie der verwendete Paketaufbau werden durch sogenannte *Protokolle* definiert. Erst das Zusammenspiel vieler Protokolle ermöglicht die heutige Kommunikation zwischen verschiedenen Anwendungsinstanzen über Knotengrenzen hinweg.

Jeder Knoten eines Netzwerks besitzt die Aufgabe, jedes eintreffende Paket entlang eines lokalen Links zum nächsten Nachbarknoten in Richtung des Ziels weiterzuleiten, dies wird auch als *Weiterleitung* (Englisch: „Forwarding“) bezeichnet. Die dafür notwendige Auswahl des nächsten Nachbarknotens erfolgt durch das *Routing* in Form der *Routingentscheidung*. Nachdem diese feststeht wird das Paket über den lokalen Link an den ausgewählten Nachbarknoten übertragen (Englisch: „Relaying“) [12]. Durch die Kombination von einzelnen Routingentscheidungen wird die sogenannte *Route* festgelegt, welche einen Weg durch das Netzwerk beschreibt und während der Weiterleitung das Paket an sein letztliches Ziel führt.

2.1.2 Namen und Adressen

Zur Bestimmung der Route benötigt das Routing stets eine eindeutige Zielbeschreibung. Dabei spielen Namen und Adressen eine wichtige Rolle. In [13] werden diese in Abschnitt 2.3 wie folgt beschrieben:

> „A name indicates what we seek. An address indicates where it is. A route indicates how to get there.“

Die Definition eines Namens wird in [12] detaillierter gegeben mit:

> “A name is a unique string, N, in some alphabet, A, that unambiguously denotes some object or denotes a statement in some language, L. The statements in L are constructed using the alphabet, A.”

Das heutige *Domain Name Systems* (DNS) [14] stellt ein Beispiel für den Einsatz von Namen dar. Es ermöglicht beispielsweise den Einsatz einfacher Strings während der Benutzung eines Webbrowsers, wodurch in für Menschen lesbarer Form die gewünschte Webseite festlegt wird. Durch die Signalisierungen von DNS wird in heutigen Netzwerken der DNS-Name auf eine Adresse abgebildet. Adressen stellen eine spezielle Form von Namen dar. In [12] wird eine Adresse beschrieben als:

> „An address is a topologically significant name, which unambiguously identifies an object or a set objects.“

Ein Netzwerkknoten stellt ein solches Objekt dar. Jeder Knoten, der ein mögliches Ziel für Pakete darstellt, muss somit über eine eindeutige Adresse identifizierbar sein und seine Adresse innerhalb der Metadaten der für ihn bestimmten Pakete gespeichert sein. Dadurch kennt jeder an der Weiterleitung beteiligte Knoten das gewünschte Ziel. Ohne diese Information sind eine Lokalisation des Ziels und Routing zu diesem Knoten nicht möglich.

2.1.3 Abstraktionsschichten der Paketweiterleitung

Das *Open Systems Interconnection* (OSI)-Modell unterteilt im Standard ISO 7498-1 [15] die paketbasierte Datenübertragung in verschiedene Abstraktionsschichten. Jeder von ihnen sind explizite Aufgaben zugeordnet, die unter Verwendung der Protokolle untergeordneter Schichten erfüllt werden.

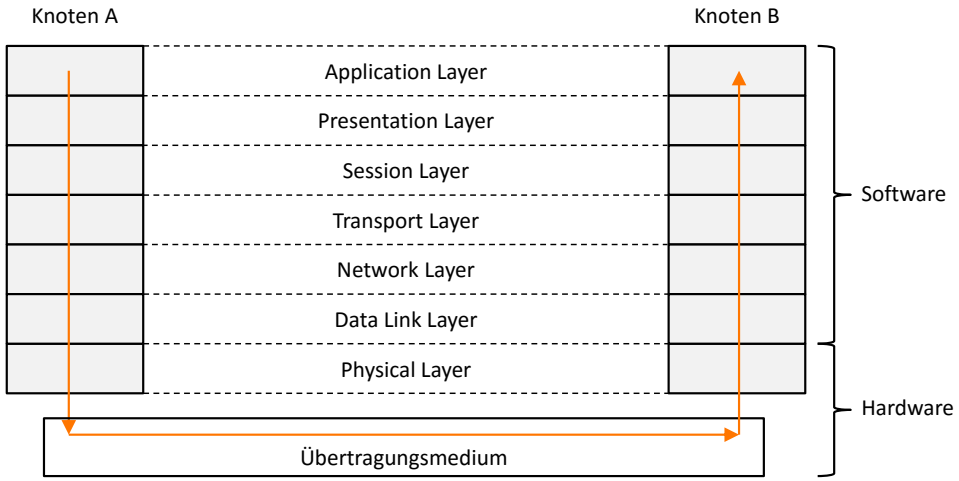

Abbildung 2.1: OSI-Modell für zwei Knoten A und B

Abbildung 2.1 stellt ein Beispiel einer Kommunikation zwischen Knoten A und B dar, die orange gefärbten Pfeile beschreiben die Übertragung von Anwendungsdaten von A nach B. Auf dem sendenden Knoten A wählt die Anwendung mit Hilfe der Programmierschnittstelle Protokolle untergeordneter Schichten explizit aus. Für jedes ausgehende Paket der Anwendung fügt ein Protokoll seine zusätzlich notwendigen Metadaten am Kopf des Pakets an. Der resultierende Aufbau der Pakete ist fest vorgegeben, sodass den Metadaten eines Protokolls von Schicht *n* stets die Metadaten eines Protokolls der untergeordneten Schicht *(n - 1)* vorangestellt werden. Auf dem Empfängerknoten B angekommen, verarbeiten die Protokolle in umgekehrter Reihenfolge die Metadaten eines Pakets. Der Vorgang endet auf Schicht 7, welche die ursprünglich versandten Anwendungsdaten der jeweiligen Anwendungsinstanz zustellt. Zu den beteiligten Schichten des OSI-Modells gehören auf Sender- und Empfängerseite:

- **Schicht 1 – *Physical Layer*:** Diese Schicht ermöglicht die physikalische Übertragung von Daten. Sie wird durch die Netzwerkschnittstellen implementiert.
- **Schicht 2 – *Data Link Layer*:** Hauptaufgabe dieser Schicht ist es, die einfache Paketweiterleitung zur Kommunikation mit einem direkten Nachbarknoten bereitzustellen. Die Protokolle dieser Schicht regeln den Zugriff auf das jeweilige Übertragungsmedium von Schicht 1. Für kabelgebundene Netzwerke wird typischerweise das Protokoll *Ethernet* entsprechend der Standards von IEEE802.3 eingesetzt, während für kabellose Netzwerke (Englisch: „wireless networks (WLAN)“) typischerweise die Standards von IEEE802.11 angewendet werden. Beide Varianten nutzen die sogenannten MAC-Adressen für die eindeutige Identifikation von Knoten eines Netzwerks. Diese werden durch den Hersteller der Netzwerkschnittstelle festgelegt und können durch die Implementierung des Protokolls von Schicht 2 ausgelesen werden.

- **Schicht 3 – *Network Layer*:** Zu den Hauptaufgaben dieser Schicht gehören das Routing sowie die dafür notwendigen Signalisierungen über Netzwerkgrenzen hinweg. Sie ermöglicht auf Basis der grundlegenden Paketweiterleitung von Schicht 2 eine Kommunikation mit entfernten Knoten. Dadurch wird eine Kommunikation zwischen jeglichen Knotenpaaren möglich. Das häufigste Protokoll dieser Schicht ist das *Internet Protokoll* (IP) in Version 4 (IPv4) [13] und 6 (IPv6) [16]. Die für IPv4/v6 notwendige Adresszuordnung wird nachfolgend in Abschnitt 2.1.5 näher erläutert. Sie ist notwendig, um den Zielknoten eines Pakets eindeutig festzulegen.
- **Schicht 4 – *Transport Layer*:** Sie wird für die Kommunikation zwischen den Anwendungsinstanzen verschiedener Knoten verwendet. Die Unterscheidung der Instanzen geschieht dabei knotenlokal über sogenannte Portnummern. Entsprechend der Vorgaben der *Internet Assigned Numbers Authority* (IANA) sind über eine Tabelle [17] die Portzuordnungen für typische Anwendungen festgelegt. Typische Protokolle der Schicht 4 sind das *Transmission Control Protocol* (TCP) [18], *User Datagram Protocol* (UDP) [19] sowie das jüngste *Stream Control Protocol* (SCTP) [20]. Die Charakteristiken der Protokolle werden dabei durch ihren vordergründigen Zweck bestimmt. Sowohl TCP als auch SCTP stellen zusätzliche Mechanismen zur Fehlererkennung und -behebung während der Kommunikation bereit. Des Weiteren unterstützen sie eine explizite Flusskontrolle pro Datenstrom. Das ermöglicht es, auftretende Engpässe im Netzwerk zu erkennen und die Datenrate zur Vermeidung von Paketverlusten zu reduzieren. Als Nachteil ergibt sich dabei ein erhöhter Signalisierungsaufwand zwischen den beteiligten Kommunikationspartnern. Hingegen stehen beim Einsatz von UDP eine möglichst hohe Datenrate und geringe Verzögerung im Vordergrund. Datenfehler werden dabei weiterhin erkannt. Im Gegensatz zu den beiden anderen Protokollen vermeidet UDP zusätzliche Signalisierungen, wodurch jedoch Übertragungsfehler im Fall von Paketverlusten nicht mehr automatisch behoben werden.
- **Schicht 5 – *Session Layer*:** Die Aufgabe dieser Schicht ist eine Dialogsteuerung zum Auf- und Abbau von Sitzungen zwischen Anwendungsinstanzen. Eine Sitzung stellt dabei eine zusätzliche Adressierungsebene dar. Des Weiteren können auf dieser Schicht Sicherungsmaßnahmen für jede Sitzung eingesetzt werden. Dadurch kann bei temporärer Unterbrechung der Kommunikation eine Sitzung nach Wiederherstellung der Kommunikation fortgesetzt und eine automatische Synchronisation der Sitzungsdaten durchgeführt werden.
- **Schicht 6 – *Presentation Layer*:** Diese Schicht definiert für systemabhängige Darstellungen von Daten entsprechende systemunabhängige Darstellungsformen, um einen korrekten Datenaustausch inhomogener Systeme zu ermöglichen. Dazu zählen unter anderem verschiedene Komprimierungs- und Verschlüsselungsvorschriften.
- **Schicht 7 – *Application Layer*:** Auf dieser Schicht befindet sich Dateneingabe und -ausgabe einer Anwendung, sodass der Zugriff durch den Benutzer ermöglicht wird.

2.1.4 Hardware zur Paketweiterleitung

Grundsätzlich besteht ein Netzwerk aus Knoten und Links. Dieser Abschnitt gibt einen Überblick über die wichtigsten Aspekte der in Netzwerken zum Einsatz kommenden Hardware. Dadurch wird das Verständnis der in Kapitel 3 erläuterten Konzeption und der darin verwendeten Unterscheidung in Hardware von Schicht 2 und 3 unterstützt.

2.1.4.1 Switches

Ein Switch dient allgemein zur Kopplung von Netzwerkknoten. Typischerweise besitzt er zwei oder mehr Netzwerkschnittstellen. An diesen befinden sich entweder Netzwerkknoten oder weitere Switches, sodass dadurch ein sogenanntes „geswitchtes“ Netzwerk gebildet wird. Dabei können verschiedene Typen von Switches eingesetzt werden [21], im Kontext dieser Arbeit werden jedoch ausschließlich Switches für Schicht 2 betrachtet. Sie implementieren die Schichten 1 und 2 des OSI-Modells. Zur Paket-

weiterleitung verwenden sie die Adressen des Protokolls von Schicht 2. Typischerweise sind das Ethernet und die darin genutzten MAC-Adressen. Ein Switch muss für eine korrekte Paketweiterleitung kontinuierlich die MAC-Adressen der angeschlossenen Geräte lernen. Erst nach Empfang des ersten Pakets eines angeschlossenen Netzwerkknotens, hat ein Switch die MAC-Adresse in seine interne Tabelle übertragen. Durch diese gespeicherten Daten kann er für eintreffende Pakete entscheiden, welcher der jeweils korrekte ausgehenden Link in Richtung des Ziels ist. Die Ausnahme bildet die Broadcast-Adresse FF:FF:FF:FF:FF:FF [22], wodurch ein Paket an alle ausgehenden Links und alle dahinter befindlichen Knoten des Netzwerks weitergeleitet wird.

Alle Knoten, welche ausschließlich über Switches für Schicht 2 miteinander gekoppelt sind, gehören derselben Broadcast-Domäne an. Dies kann ebenfalls auf zwei direkt benachbarte Router zutreffen, welche über einen direkten Link miteinander verbunden sind. Der Begriff der Broadcast-Domäne bezieht sich auf die spezifischen Broadcast-Nachrichten eines Protokolls von Schicht 2. Wird dabei *Ethernet* eingesetzt und ein Knoten sendet ein sogenanntes *Ethernet Frame* an die Broadcast-Adresse FF:FF:FF:FF:FF:FF, wird dieses an alle anderen Knoten der Domäne übertragen. Gestoppt wird ein solches Frame ausschließlich von Routern (siehe Abschnitt 5.1 in [23]).

2.1.4.2 Router

Ein Netzwerkknoten kann verschiedene Aufgaben besitzen. Es kann sich dabei entweder um einen Endknoten oder ein Bestandteil der Netzwerkinfrastruktur (oder auch beides) handeln. Im ersten Fall ist es beispielweise ein Bürorechner, welcher typischerweise nur eine Netzwerkschnittstelle besitzt. Auf ihm werden Anwendungsinstanzen ausgeführt, welche über das Netzwerk mit anderen Instanzen kommunizieren. Im zweiten Fall ist der Knoten typischerweise ein sogenannter Router, welcher über mehrere Netzwerkschnittstellen verfügt, über die er jeweils ein geswitchtes Netzwerk erreicht. In [24] ist in Abschnitt 1.2 zu lesen:

> „A router's functions are to read the destination address marked in an incoming IP packet, to consult its internal information to identify an outgoing link to which the packet is to be forwarded, and then to forward the packet."

Ein Router wertet im Gegensatz zu Switches für eintreffende Pakete die Metadaten des Protokolls von Schicht 3 aus, um die jeweils gewünschte Zieladresse zu ermitteln und das Paket dementsprechend weiterzuleiten. Er erfüllt somit die Rolle des Bindegliedes zwischen zwei oder mehr angeschlossenen, geswitchten Netzwerken. Nähere Details zur Auswahl des jeweils nächsten Knotens zur Weiterleitung eines Pakets in Richtung seines Ziels werden in Abschnitt 2.1.6 beschrieben. Des Weiteren geht Abschnitt 2.1.7 näher auf die dafür notwendigen Tabellen ein, welche ein Router lokal als Datenbasis für seine Entscheidungen speichern muss.

2.1.5 Identifikation von Netzwerkschnittstellen eines Knotens

Ein Router muss für seine Entscheidungen sowohl das Ziel eines Pakets als auch die möglichen Zwischenknoten eindeutig identifizieren können. Zu diesem Zweck werden in heutigen Netzwerken typischerweise Adressen des Internet Protokolls verwendet. Entsprechend den Abschnitten 2.3 in [13] und 2.1 in [25] werden dabei sowohl IPv4- als auch IPv6-Adressen für jede Netzwerkschnittstelle vergeben, sodass ein Knoten eine oder mehrere logische IP-Adressen zugeordnet sein können, über die er identifizierbar ist.

2.1.5.1 Aufbau von IP-Adressen

Eine IPv4-Adresse besteht aus 32 Bits, welche als 4 Oktetten interpretiert werden. Sie werden typischerweise als „A.B.C.D" dargestellt. Jeder Buchstabe steht für eine dezimale Zahl zwischen 0 und 255. Dagegen besteht eine IPv6-Adresse aus 128 Bits, welche als 8 Doppeloktetten mit einer Darstellung der

Form „A:B:C:D:E:F:G:H“ interpretiert werden. Jeder Buchstabe steht für eine hexadezimale Zahl zwischen 0x0 und 0xffff.

IP-Adressen enthalten stets einen netzwerk- und einen schnittstellenspezifischen Teil. Zur Abgrenzung beider Teile wird neben der eigentlichen Adresse für IPv4 eine Netzmaske und für IPv6 eine Präfixlänge angegeben. Somit ist die Anzahl der Bits der Adresse für den Netzwerkteil festgelegt. Beispielsweise gehört eine IPv4-Adresse „192.168.2.5“ mit der Netzmaske „255.255.0.0“ dem Netzwerk „192.168.0.0“ an. Für eine IPv6-Adresse „fc00:1:2:3:8:8:8:8“ mit einer Präfixlenge von 64 ergibt sich das Netzwerk „fc00:1:2:3:0:0:0:0“. Durch diese Unterscheidung kann das Ziel einer Route oder eines Pakets durch eine schnittstellenübergreifende Netzwerkidentifikation verallgemeinert werden.

2.1.5.2 Private Adressbereiche

Neben den global gültigen IP-Adressen gibt es sowohl für Version 4 als auch Version 6 zusätzliche Adressbereiche, welche als privat gelten. Die nachfolgenden Tabellen geben einen Überblick über die privaten IPv4- und IPv6-Adressbereiche [26].

Netzwerk	Netzmaske	Erlaubte Knotenadressen	Verwaltung und Vergabe
10.0.0.0	255.0.0.0	10.0.0.1 bis 10.255.255.254	lokal
172.16.0.0	255.240.0.0	172.16.0.1 bis 172.31.255.254	lokal
192.168.0.0	255.255.0.0	192.168.0.1 bis 192.168.255.254	lokal

Tabelle 2.1: Private Adressbereiche für IPv4

Tabelle 2.1 zeigt die privaten Adressbereiche für IPv4. Diese Adressen können für private Netzwerke verwendet werden. Ihre Vergabe kann für das jeweilige Netzwerk beliebig durch den Netzwerkadministrator konfiguriert werden. Pakete mit einer Zielbeschreibung aus einem dieser Bereiche werden von keinem Router des Internets weitergeleitet [27].

Länge (in Bits)	Inhalt
8	Netzwerknummer
24	Lokale Adresse

Tabelle 2.2: Format einer IPv4-Adresse aus dem größten privaten Adressbereich

Tabelle 2.2 zeigt den Aufbau einer IP-Adresse des Netzwerks 10.0.0.0, es bietet den größten zusammenhängenden Adressbereich für private Netzwerke.

Netzwerk	Präfixlänge (in Bits)	Erlaubte Knotenadressen	Verwaltung und Vergabe
fc00:0:0:0:0:0:0:0	7	fc00:0:0:0:0:0:0:1 bis fcff:ffff:ffff:ffff:ffff:ffff:ffff:ffff	global zugewiesen
fd00:0:0:0:0:0:0:0	7	fd00:0:0:0:0:0:0:1 bis fdff:ffff:ffff:ffff:ffff:ffff:ffff:ffff	lokal

Tabelle 2.3: Private Adressbereiche für IPv6

Tabelle 2.3 gibt einen Überblick über die sogenannten *Unique-Local-Unicast*-Adressen von IPv6. Sie ähneln den privaten IPv4-Adressen. Pakete mit einer solchen Zieladresse werden, analog zu den privaten IPv4-Adressen, von Routern des Internets nicht weitergeleitet [28]. Insbesondere die Adressen des Netzwerks „fd00:0:0:0:0:0:0:0“ können lokal von einem Netzadministrator verwaltet werden, sodass sie, ohne äußere Vorgaben, den jeweils vorhandenen Netzwerkknoten zugewiesen werden.

Länge (in Bits)	Inhalt
7	Präfix
1	L
40	Globale ID
16	Subnetz-ID
64	Schnittstellen-ID

Tabelle 2.4: Format einer IPv6-Adresse aus dem privaten Adressbereich

Wie in Tabelle 2.4 ersichtlich, folgen den 7 Bits des Adresspräfixes zusätzliche Werte zur Klassifikation einer Adresse. IPv6-Adressen bestehen aus folgenden Teilen [28]:

- **Präfix:** Im Allgemeinen wird hierdurch das Netzwerk festgelegt. Die Präfixe „fc00“ und „fd00“ ordnen die jeweilige Adresse der Klasse der *Unique Local Unicast* Adressen zu.
- **L:** Dieses Bit ist für das Netzwerk fd00:0:0:0:0:0:0:0 stets mit dem Wert 1 festgelegt.
- **Globale ID:** Für jedes Netzwerk wird ein global eindeutiger Wert verwendet, um für verschiedene Standorte möglichst disjunkte Adressbereiche zu verwenden.
- **Subnetz-ID:** Dieser Wert ermöglicht eine zusätzliche Unterteilung der Adressen pro Standort. Er steht für ein Teilnetzwerk.
- **Schnittstellen-ID:** Dieser Wert wird pro Netzwerkschnittstelle vergeben. Es kann sich dabei beispielsweise um eine MAC-Adresse handeln, insofern Ethernet auf Schicht 2 zum Einsatz kommt. Genauere Details dazu sind dem Abschnitt 2.5.1 in [25] zu entnehmen.

2.1.5.3 Adressvergabe

Neben der statischen Adressvergabe durch einen Netzwerkadministrator existieren sowohl für IPv4 als auch IPv6 Mechanismen, mit deren Hilfe hinzukommenden Netzwerkknoten automatisch eine IP-Adresse durch einen zentralen Knoten zugewiesen werden kann. Der Grundgedanke aller Protokolle ist es, ohne zusätzlichen Aufwand dynamisch für jeden hinzukommenden und entfernten Knoten eine Konfiguration zu ermitteln. Es sollen alle Informationen signalisiert werden, welche für die aktive Teilnahme am Netzwerk notwendig sind. Dazu zählen insbesondere eine zugewiesene IP-Adresse sowie die IP-Adresse eines Gateway-Knotens.

Die am häufigsten eingesetzte Lösung für lokale Netzwerke stellt das *Dynamic Host Configuration Protocol* (DHCP) als Nachfolger des *Bootstrap Protocols* (BOOTP) [29] dar. Es ist sowohl für IPv4 [30] als auch für IPv6 [31] verfügbar. Das Protokoll beruht auf einer Client-Server-Kommunikation, bei der auf Basis der Signalisierungen ein Knoten explizit um Zuweisung einer IP-Adresse durch den jeweiligen DHCP-Server bitten kann. Neben weiteren, für diese Arbeit nicht relevanten, Daten erhält der Knoten folgende Informationen:

- **bei IPv4:**
  - IP-Adresse der Netzwerkschnittstelle
  - Netzwerkmaske der Netzwerkschnittstelle
  - IP-Adresse des Gateway-Knotens der zugehörigen Broadcast-Domäne
- **bei IPv6:**
  - IP-Adresse der Netzwerkschnittstelle
  - Präfixlänge der Netzwerkschnittstelle
  - IP-Adresse des Gateway-Knotens der zugehörigen Broadcast-Domäne

DHCP ermöglicht eine zentrale Verwaltung der Konfigurationsparameter für die Knoten eines Netzwerks. Bei Topologieänderungen kann dadurch konzentriert an einem Punkt des Netzwerks die Adressvergabe an die neue Netzwerkstruktur angepasst werden. Für dynamische Netzzugänge über einen *Internet Service Provider* (ISP) existieren separate Lösungen. Darunter zählen beispielsweise das *Point-to-Point Protocol* (PPP) [32] und das *Point-to-Point Protocol over Ethernet* (PPPoE) [33]. Beide basieren, ähnlich DHCP, auf einer Client-Server-Kommunikation und weisen jedem anfragenden Knoten eine explizite IP-Adresse zu. Die Konfiguration der Adressen geschieht in diesem Fall zentral durch den ISP.

Des Weiteren kann für IPv6 die Adressvergabe mittels *Stateless Address Autoconfiguration* (SLAAC) [34] geschehen. Im Gegensatz zu den vorherigen Lösungen basiert SLAAC auf einer *Peer-to-Peer*(P2P)-Kommunikation, bei der alle Kommunikationspartner gleichberechtigt sind. Dabei wird das *Neighbor Discovery Protocol* (NDP) [35] verwendet, um einen gemeinsamen Konsens bezüglich der Adressvergabe unter den Routern des lokalen Netzwerks herzustellen. Des Weiteren werden durch die Signalisierung von NDP vorhandene Router des Netzwerks erkannt. Im Gegensatz zu DHCP wird insbesondere die Adressvergabe und Routererkennung realisiert. Der volle Signalisierungsumfang von DHCP wird nicht unterstützt, sodass in heutigen Netzwerken SLAAC häufig mit DHCP kombiniert wird.

Als typisches Routingprotokoll für heutige Netzwerke ist *Open Shortest Path First* (OSPF) ebenfalls im Kontext der Adressvergabe interessant. Ein OSPF-Router verwendet als zentrale Adresse die sogenannte *Router-ID*. Auf Basis dieser ID können Signalisierungsnachrichten ihrem Sender zugeordnet werden. Sie entspricht bei OSPFv2 [36] und OSPFv3 [6] einer IPv4- bzw. IPv6-Adresse und wird vom Netzwerkoperator vergeben. Für OSPFv3 wird die Vergabe mit folgenden Worten näher beschrieben:

> "Possible Router ID assignment procedures for IPv6 include: a) assign the IPv6 Router ID as one of the router's IPv4 addresses or b) assign IPv6 Router IDs through some local administrative procedure (similar to procedures used by manufacturers to assign product serial numbers)" (Abschnitt C.1 in [6])

Daraus lässt sich eine manuelle Vergabe der IP-Adressen für Router bei Version 2 als auch der aktuelle Version 3 von OSPF ableiten. Weitere Details zu OSPF werden in Abschnitt 2.1.8.2 gegeben.

2.1.6 Routingentscheidungen

Routing ist einer der wesentlichen Faktoren für die Gesamtperformanz eines Netzwerks. Es besteht aus Einzelentscheidungen des sogenannten Routingalgorithmus[1], der die lokal bekannten Daten über die Topologie des Netzwerks, die sogenannten Routingdaten, als Eingabe nutzt. Erst die Kombination von mehreren dieser Entscheidungen ergibt eine vollständige Route von der Quelle zum Ziel einer Übertragung. Da die Kapazitäten aller Links jedoch begrenzt sind, kann eine einzelne suboptimale Routingentscheidung bereits zu Engpässen oder Überlastungen im Netzwerk führen. Pakete werden entweder verzögert übertragen oder bei Überlastung eines Links unmittelbar verworfen.

Die einfachste Möglichkeit zur Bestimmung von Routingentscheidungen ist eine manuelle Konfiguration. Daraus ergibt sich ein sogenanntes statisches Routing, welches für die Laufzeit des Netzwerks

[1] Während in der Literatur unter dem Begriff des Routingalgorithmus typischerweise sowohl die Routenberechnung als auch die Routingentscheidung zusammengefasst werden, wird in dieser Arbeit der Begriff des Routingalgorithmus insbesondere für die Ermittlung der Routingentscheidung verwendet. Die Routenberechnung wird als ein vorheriger, notwendiger Schritt verstanden, der sowohl unmittelbar vor der tatsächlichen Entscheidung als auch zu festgelegten Zeitpunkten durchgeführt werden kann.

konstante Ergebnisse liefert. Sie sind sowohl von der aktuellen Topologie als auch der Netzwerkauslastung unabhängig. In heutigen Netzwerken wird statt des statischen Routings eher dynamisches Routing eingesetzt, wobei Routingentscheidungen durch verschiedene Faktoren beeinflusst werden. Sowohl die Ziele der Netzwerknutzer als auch des jeweiligen Netzwerkbetreibers stehen im Vordergrund[2]. Aus Sicht des Nutzers soll das Routing stets Pfade wählen, auf denen Pakete möglichst mit hoher Datenraten und schnell von der Quelle zum Ziel gelangen. Dabei spielt Fairness zwischen den Datenströmen einzelner Nutzer eine wichtige Rolle. Niemand darf exklusiv die Ressourcen des Netzwerks verwendet. Dagegen orientiert sich der Netzwerkbetreiber eher an wirtschaftlichen Faktoren (Kostenminimierung und Nutzenmaximierung). Er ist daran interessiert, dass möglichst effiziente Pfade durch das Routing ausgewählt werden. Dadurch sollen möglichst viele Nutzer einen akzeptablen Service im Rahmen der zur Verfügung stehenden Ressourcen erhalten. Der einzelne Nutzer und das für ihn beste Routing stehen dabei nicht im Vordergrund.

2.1.6.1 Metriken und Routingstrategien

Die verfügbaren Routingansätze können anhand ihrer Strategie klassifiziert werden [24]. Am häufigsten wird dabei *Shortest Path Routing* in heutigen Netzen verwendet, welches insbesondere im IPv4/IPv6-basierten Internet eingesetzt wird. Dabei steht das Ziel im Vordergrund, Pakete möglichst entlang des kürzesten Pfads zu leiten. Die Länge eines Pfads kann sich dabei aus der Hop-Distanz, gemessen an der Anzahl zu passierender Knoten, ergeben. Alternativ kann die resultierende Verzögerung einer Route, gemessen an der notwendigen Zeit zur Übertragung der Pakete, als Kriterium für die Distanz verwendet werden. Die sogenannte Kostenfunktion abstrahiert die gewählte Art der Berechnung und ordnet einer Route einen skalaren Wert zu, welcher die Kosten einer Nutzung ausdrückt. Dabei können sowohl einals auch mehrdimensionale Eingaben verwendet werden. Sie werden als die *Metrik* des Routings bezeichnet dar. Entscheidend bei der Bestimmung der Gesamtkosten einer Route ist die Vorschrift zur Kombination der Kosten von mehreren Links:

- **additive Kosten:** Die Gesamtkosten ergeben sich als Summe der Einzelkosten. Beispiele dafür sind die Anzahl zu passierender Knoten oder die Übertragungsverzögerung entlang einer Route.
- **konkave Kosten:** In diesem Fall wird das Minimum der Einzelkosten als Ergebnis verwendet. Ein Beispiel dafür ist die verfügbare Datenrate entlang einer Route.
- **multiplikative Kosten:** Die Gesamtkosten ergeben sich aus der Multiplikation der Einzelkosten. Ein Beispiel dafür ist die Wahrscheinlichkeit für Paketverluste entlang einer Route.

Für *Shortest Path Routing* ergeben sich die Gesamtkosten stets aus der Summe der Einzelkosten der verwendeten Links. Im Gegensatz dazu stehen beim sogenannten *Widest Path Routing* die konkaven Kosten in Form der verfügbaren Datenrate einzelner Links im Vordergrund. Der Routingalgorithmus wählt dadurch jeweils den „breitesten“ (Englisch: „widest“) Pfad durch das Netzwerk bevorzugt aus.

2.1.6.2 Verteilung von Routenberechnungen

Zur Berechnung des kürzesten und auch des breitesten Pfads werden typischerweise entweder der *Bellman-Ford* oder der *Dijkstra*-Algorithmus eingesetzt [24]. Beide Algorithmen benötigen für eine knotenlokale Berechnung die Routingdaten fremder Router, welche die verfügbare Topologie entfernter Netzabschnitte beschreiben. Beide Algorithmen können dabei sowohl auf vollständigen als auch begrenzten Routingdaten arbeiten. Es existieren zwei typische Implementierungsvarianten:

- **Zentrale Routinginstanz und -entscheidung:** Es existiert eine zuständige Kontrollinstanz im Netzwerk, welche die gesamte Netzwerktopologie kennt und die Gesamtroute festlegt. Den Zwischenknoten wird diese Entscheidung mit Hilfe einer expliziten Signalisierung, beispiels-

[2] Dies wird in Abschnitt 2.1 in [22] ausführlich diskutiert.

weise durch zusätzliche Metadaten in den Paketen, mitgeteilt. Die Zwischenknoten degenerieren dadurch zu einer einfachen Weiterleitungsinstanz. Quellbasiertes Routing, auch als *Source Routing* [13] bezeichnet, ist ein Beispiel dieses Ansatzes. Sein Vorteil besteht in der Genauigkeit der Eingabedaten. Da alle Daten über die Topologie des Netzwerks auf einem Knoten lokal zur Verfügung stehen, wird eine Routingentscheidung stets auf Basis einer vollständigen Sicht auf das Netzwerk getroffen. Nachteilig sind dabei die notwendigen Signalisierungen von Nachbarschaftsbeziehungen zwischen allen Knoten. Da jeder Knoten einen potentiellen Startpunkt für Übertragungen darstellt, muss jeder Knoten eine lokale globale Sicht über das Netzwerk aufbauen.

- **Verteilte Routinginstanzen und -entscheidungen:** Die Routingdaten sind im Netzwerk verteilt gespeichert, sodass ein verteiltes Routing angewandt werden kann. Knoten übernehmen sowohl die Aufgabe der Routenberechnung als auch der Weiterleitung von Paketen. Beispielsweise wird im heutigen Internet das sogenannte *Hop-by-Hop*-Routing angewandt, bei dem jeder Knoten über den Link zum jeweils nächsten Knoten in Richtung des jeweiligen Paketziels entscheidet. Das resultierende Gesamtrouting besteht somit aus den Einzelentscheidungen der beteiligten Zwischenknoten. Die beiden Abschnitte 2.1.8.2 und 2.1.8.3 beschreiben zwei dieser Ansätze für heutige Netzwerke. Alternativ dazu gibt es Ansätze, welche sogenannte Pfadsegmente einsetzen. Das resultierende Gesamtrouting wird in diesem Fall durch spezielle Zwischenknoten bestimmt, welche jeweils mehrere aufeinanderfolgende Links zu einem Routensegment kombinieren. Ein solches Routing wird beispielsweise im Abschnitt 2.3.4 angewendet. Im Vergleich zum zentralen Routing liegt der Vorteil des verteilten Routings in der typischerweise begrenzten Sicht auf das Netzwerk. Ein Knoten kennt nicht mehr alle topologischen Details entfernter Netzabschnitte, wodurch der lokal verursachte Speicher- und Signalisierungsaufwand niedrig gehalten werden kann. Des Weiteren ist durch die Verteilung von Berechnungen im Vergleich zum zentralen Routing ein schnelleres lokales Rerouting im Fall von spontanen Ausfällen möglich. Nachteilig für ein verteiltes Routing kann der reduzierte Detailgrad der bekannten Topologie sein, wodurch suboptimale Routingentscheidungen verursacht werden können. In diesem Fall kann die ermittelte Route beispielsweise länger als die optimale Lösung ausfallen.

Unabhängig von der Verteilung von Routenberechnungen können durch Signalisierungsverzögerungen die Daten eines Routers veraltet sein, was sich in suboptimalen Routingentscheidungen äußern kann.

2.1.6.3 Berechnungszeitpunkt

Unabhängig von der Platzierung von Routinginstanzen können Routingentscheidungen sowohl vor als auch nach dem Eintreffen einer Routinganfrage berechnet werden. Im ersten Fall spricht man vom proaktiven Routing. Typischerweise sorgen periodische Signalisierungen dafür, dass bereits vor dem Eintreffen von Anwendungsdaten die notwendigen Daten über die Topologie des Netzwerks in aktueller Form vorliegen. Im Gegensatz dazu werden beim Einsatz der zweiten Variante erst bei Eintreffen einer Routinganfrage die für die Berechnung notwendigen Daten ermittelt, diese Form wird als reaktives Routing bezeichnet.

	Proaktives Routing	Reaktives Routing
Laufzeit	• Route liegt jederzeit vor	• Route liegt erst nach Signalisierungen vor
Vorteile	• konstante Verzögerung der Paketweiterleitung • geringe Verzögerung	• keine periodische Signalisierung und dadurch nur Verbrauch von Netzwerkressourcen, wenn Routinganfragen vorliegen • Verwendung von stets aktuellen Routingdaten • ausschließliche Speicherung von genutzten Routen
Nachteile	• periodische Aktualisierungen verbrauchen Netzkapazitäten • verzögerte Reaktion auf Topologieänderungen • Speicherung von ungenutzten Routen	• höhere Verzögerung im Vergleich zu proaktivem Routing • variable Verzögerungen durch Signalisierungen

Tabelle 2.5: Vergleich zwischen proaktivem und reaktivem Routing

Tabelle 2.5 gibt einen Überblick über die signifikanten Unterschiede beider Routingtypen. Während proaktives Routing sowohl in kabelgebundenen als auch kabellosen Netzwerken eingesetzt wird, spielt reaktives Routing eher für kabellose adhoc-Netzwerke eine Rolle. Sie werden insbesondere in Sensornetzwerken verwendet, um nur bei Bedarf die Batterien der Sensorknoten durch zusätzliche Signalisierungen zu belasten. Eine hybride Lösung ist ebenfalls möglich und wird beispielweise für adhoc-Netzwerke angewandt. Dabei werden proaktive Berechnungen innerhalb von Teilnetzwerken angewandt und nur für Übertragungen über Netzwerkgrenzen hinweg werden reaktiv neue Pfade ermittelt.

2.1.7 Routenspeicherung

Ein Knoten muss typischerweise die Pakete von verschiedenen Sendeknoten und Anwendungsinstanzen weiterleiten. Demzufolge müssen sehr viele Weiterleitungen und die damit verbundenen Routingentscheidungen in sehr kurzer Zeit ausgeführt werden können. Dabei ist die benötigte Zeit von einzelnen Entscheidungen maßgebend für die Performanz des Gesamtsystems. Damit ein Knoten nicht für jedes Paket eine komplette Neuberechnung durchführen muss, werden in heutigen Implementierungen zusätzlich Tabellen eingesetzt. Sie stellen einen Speicher dar, der bereits durchgeführte Berechnungen für zukünftige Pakete zwischenspeichert. Grundsätzlich wird dabei zwischen Routing- und Weiterleitungstabelle unterschieden. Erstere wird insbesondere für verteiltes Routing eingesetzt, sodass eine einmal durchgeführte Routenberechnung für nachfolgende Pakete und den damit notwendigen Routingentscheidungen wiederverwendet werden kann. Eine solche Tabelle ist bei quellbasiertem Routing nicht notwendig, da in diesem Fall die gesamte Route durch die Quelle festlegt und für nachfolgende Router typischerweise innerhalb des Pakets abspeichert wird. Die Weiterleitungstabelle bietet hingegen alle notwendigen Daten für die allgemeine Paketweiterleitung (siehe Abschnitt 2.1.1), sie enthält neben den Daten der Routingtabelle auch die notwendigen Informationen für die Übergabe eines Pakets an den durch das Routing ausgewählten Nachbarnoten.

2.1.7.1 Routingtabellen

Zuerst wird die Routingtabelle näher betrachtet, sie ist oft auch unter der Bezeichnung *Routing Information Base* (RIB) zu finden. Jeder Tabelleneintrag steht für eine verfügbare Route, der Inhalt kann dabei entweder unveränderlich (statisches Routing) oder veränderlich (dynamisches Routing) sein.

Ziel		Nächster Knoten (Gateway)	Metrik
Knoten- /Netzwerkadresse	**Netzwerkmaske**	**Knotenadresse**	**Hop-Distanz**
192.168.23.0	255.255.255.0	192.168.1.1	1
10.0.0.0	255.0.0.0	192.168.1.1	1

Tabelle 2.6: Beispiel einer IPv4-Routingtabelle

Ein Eintrag der Routingtabelle, wie sie in Tabelle 2.6 beispielsweise für IPv4 dargestellt ist, beinhaltet folgende Komponenten[3]:

- **Ziel:** Das Ziel der jeweiligen Route kann entweder eine einzelne Adresse oder ein Netzwerk darstellen. Im letzteren Fall kann die Route für verschiedene Zielknoten verwendet werden. Typischerweise werden Adressen von Schicht 3 verwendet. Für IPv4 wird zusätzlich die Netzmaske und für IPv6 die Präfixlänge zur Zieladresse gespeichert.
- **Nächster Knoten:** Die Adresse des nächsten Knotens[4] entlang der jeweiligen Route.
- **Metrik:** Die Metrik stellt eine Kostenrepräsentation dar. Es kann eine Priorisierung zwischen Routen zum gleichen Ziel festgelegt werden. In heutigen Netzwerken wird dafür typischerweise die Anzahl von Knoten zum jeweiligen Ziel einbezogen. Zusätzlich kann die Metrik implementierungsspezifisch durch Werte für Datenrate und Verzögerung erweitert werden.

Im Kontext dieser Arbeit steht insbesondere das dynamische Routing im Vordergrund, bei dem die Inhalte der Routingtabellen typischerweise durch spezielle Protokolle für Schicht 3 und darüber bestimmt und seine periodisch ablaufenden Signalisierungen kontinuierlich aktualisiert werden. Sollten auf einem Knoten mehrere dieser Protokolle aktiv sein, erstellt jedes seine eigene Routingtabelle. Weitere Details zur Erstellung und Aktualisierung von Routingtabellen sind in Abschnitt 2.1.8 zu finden.

2.1.7.2 Weiterleitungstabellen

Zusätzlich zu den Daten der Routingtabelle benötigt ein Knoten ebenfalls Kenntnis über den ausgehenden Link, über den ein Paket an den nächsten Knoten weitergeleitet werden muss. Diese Lücke füllt in heutigen Implementierungen die sogenannte Weiterleitungstabelle. Sie ist innerhalb der Literatur auch unter der Bezeichnung *Forwarding Information Base* (FIB) zu finden.

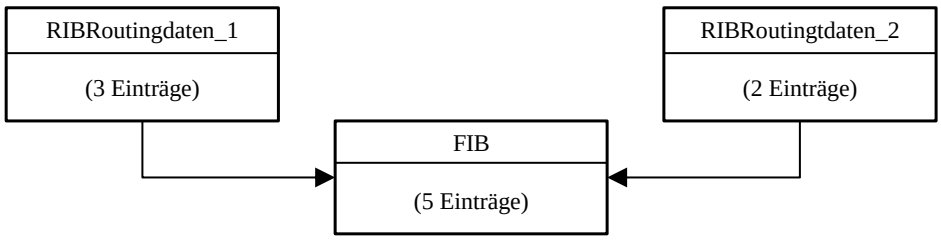

Abbildung 2.2: Bildung der Forwarding Information Base

Wie in Abbildung 2.2 ersichtlich, wird eine FIB aus den Daten aller lokal vorhandenen Routingtabellen erstellt. Zusätzlich werden die lokal vorliegenden Daten über die Nachbarschaftsbeziehungen einbezogen, sodass die Adressen von Schicht 3 auf Adressen von Schicht 2 abgebildet werden. Dabei kommt

[3] Siehe Abschnitt 14.1.4 in [22].
[4] Der nächste Knoten wird in heutigen Betriebssystemen oft als „Gateway" bezeichnet. Beispielsweise ist dies bei Eingabe des Befehls „route print" unter MS Windows und durch den Befehl „route" unter Linux ersichtlich.

typischerweise das *Address Resolution Protocol* (ARP) [37] zum Einsatz. Jeder Eintrag einer FIB beinhaltet für den jeweils nächsten Knoten ebenfalls seine Adresse des Protokolls von Schicht 2 sowie die ausgehende Netzwerkschnittstelle zu ihm.

Ziel		**Nächster Knoten (Gateway)**	**Metrik**	**Schnittstelle**	**Adresse von Schicht 2**
Knoten- / Netzadresse	**Netzmaske**	**Knotenadresse**	**Hop-Distanz**	**Name**	**MAC-Adresse**
192.168.23.0	255.255.255.0	192.168.1.1	1	eth0	00:1a:a0:1b:9d:8c
10.0.0.0	255.0.0.0	192.168.1.1	1	eth0	00:1a:a0:1b:9d:8c

Tabelle 2.7: Beispiel einer IPv4-Weiterleitungstabelle bei Verwendung von Ethernet

Wird beispielsweise Ethernet auf Schicht 2 verwendet, enthält jeder FIB-Eintrag die MAC-Adresse des jeweiligen Nachbarknotens, der den nächsten Schritt in Richtung des Zielknotens darstellt. Tabelle 2.7 zeigt ein Beispiel einer solchen Weiterleitungstabelle, zusätzlich zu den Daten der Routingtabelle sind darin der Name der Schnittstelle und die MAC-Adresse zum jeweils nächsten Knoten zu sehen.

2.1.8 Routingprotokolle heutiger Netzwerke

Ein typisches Routing im Internet und in lokalen Firmennetzwerken arbeitet dynamisch, verteilt und proaktiv. Die Routingtabellen werden jederzeit aktuell gehalten, sodass Routinganfragen sofort, ohne zusätzliche Wartezeiten für ausstehende Signalisierungen, beantwortet werden können. Die Implementierung eines Routings geschieht dabei auf Basis eines festgelegten Routingprotokolls. Dieses wird für die Router-zu-Router Signalisierungen eingesetzt, wobei als Router die Knoten zu verstehen sind, welche Routingentscheidungen treffen. Durch den Einsatz eines Routingprotokolls werden zwischen den Routern die Routingdaten verteilt. Aus diesen Daten erstellt jeder Router seine lokale Routingtabelle. Ein knotenübergreifendes Routingprotokoll definiert:

- **Prozesse und Signalisierungsdaten:** Es werden Prozesse definiert, welche typischerweise Ereignisse und Reaktionen darauf definieren, sodass Daten zwischen Knoten durch Signalisierungsnachrichten entweder einmalig (aufgrund des Eintreffens eines Ereignisses) oder periodisch (nach Ablauf eines Intervalls) synchronisiert werden. Des Weiteren wird festgelegt, wie lokale Daten in Paketen repräsentiert werden, sodass der Empfänger diese in ihrer ursprünglichen Form wiederherstellen und lokal weitere Prozesse auslösen kann.
- **Nachrichtenformate:** Für jede Signalisierung werden die eingesetzten Nachrichtenformate definiert, wodurch die Reihenfolge sowie die Kodierungen der zu übertragenden Datenelemente festgelegt sind.

2.1.8.1 Klassifikation

Das Internet ist in sogenannte *autonome Systeme* (AS) [38] unterteilt. Richtlinien für diese Unterteilung in heutigen IP-basierten Netzwerken sind in [39] festgelegt. Darin wird ein AS definiert als:

> "An AS is a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy."

Ein AS umfasst folglich eine administrative Domäne [40], welche eine einheitliche Routingpolitik eines Netzwerkbetreibers implementiert. Innerhalb eines AS wird dabei von jedem Router das gleiche Routingprotokoll eingesetzt, mit dessen Hilfe Routingdaten über das jeweilige AS ausgetauscht werden. Durch die daraus abgeleiteten Routingtabellen sind interne Routen auf jedem Router innerhalb des AS bekannt.

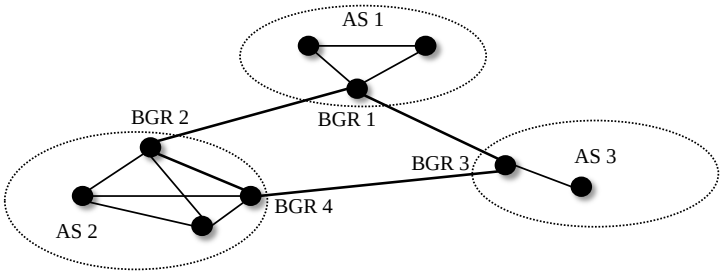

Abbildung 2.3: Beispiel eines Netzwerks mit drei autonomen Systemen

Abbildung 2.3 zeigt ein einfaches Netzwerk mit drei autonomen Systemen. In jedem befinden sich AS-interne Router, welche über ihr internes Routingprotokoll miteinander kommunizieren. Für eine Übertragung zu AS-fremden Netzwerkabschnitten ist ein weiteres Routingprotokoll notwendig. Es realisiert die Verknüpfung zwischen den autonomen Systemen auf Basis der *Border Gateway Router* (BGR), welche sich auf der Grenze ihres zugehörigen AS befinden. Jeder BGR unterstützt dabei typischerweise mehr als ein Routingprotokoll und besitzt neben den Daten des Intra-AS-Routings zusätzliche Routingdaten von den zu ihm fremden AS. Dadurch ist er in der Lage, Routen zu diesen Netzwerkabschnitten, das sogenannte Inter-AS-Routing, zu bestimmen. Entsprechend Abschnitt 2.1.7.2 wird in diesem Fall seine lokale FIB aus den RIB-Instanzen aller lokal ablaufenden Routingprotokolle abgeleitet.

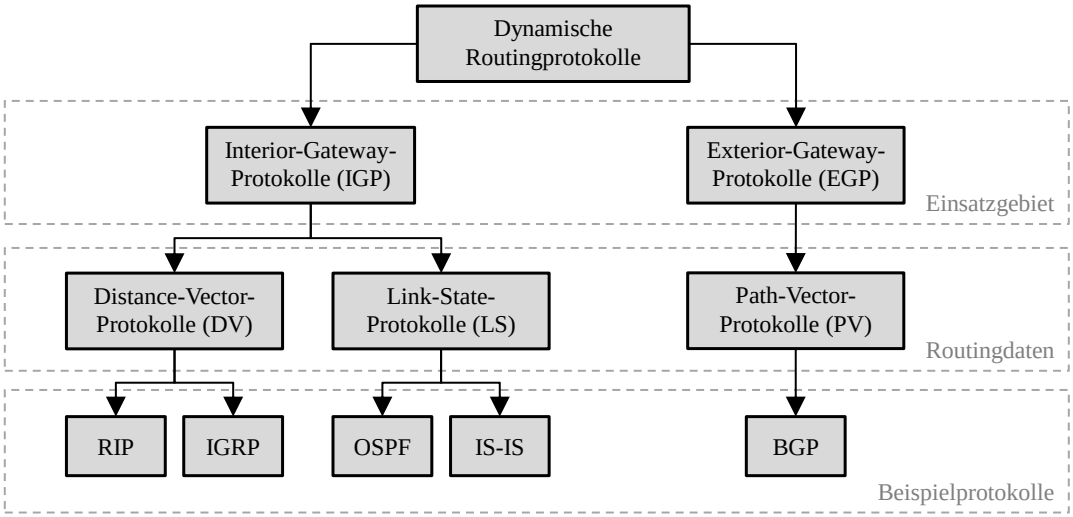

Abbildung 2.4: Klassifikation anhand des Einsatzgebietes und der Art von Routingdaten

Abbildung 2.4 zeigt eine Klassifikation der Routingprotokolle anhand ihres Einsatzgebietes. Man unterscheidet zwischen der Klasse der *Interior-Gateway-Protokolle* (IGP) und der *Exterior-Gateway-Protokolle* (EGP). Erstere werden innerhalb eines AS eingesetzt und letztere werden für das Inter-AS-Routing verwendet. Dadurch wird eine zweistufige Hierarchie im Internetrouting angewandt, auf die im Folgenden näher eingegangen wird.

2.1.8.2 Intra-AS-Routing

Für das Intra-AS-Routing wurde das *Routing Information Protocol* (RIP) [41] entwickelt, es gehört zur Klasse der *Distance-Vector*-Protokolle. Diese haben gemeinsam, dass die beteiligten Router untereinander Tabellen austauschen, welche die jeweils lokal bekannten kürzesten Routen inklusive ihrer Kosten zu möglichen Zielen des Netzwerks beschreiben. Diese Daten werden periodisch zwischen den Nachbarn ausgetauscht, sodass letztlich jeder Router alle Knoten des Netzwerks erreichen kann. Dem Prozess liegt der *Bellman-Ford*-Routingalgorithmus in einer verteilt arbeitenden Variante zu Grunde[5]. Der

[5] Siehe Abschnitt 2.2.2 in [22]

Nachteil dieses Ansatzes besteht in der erhöhten Konvergenzzeit bei Topologieänderungen sowie dem Hang zur Bildung von sogenannten *Routingschleifen*, wodurch Pakete bei ihrer Weiterleitung einen oder mehrere Router mindestens zweimalig passieren. Weitere Details dazu sind in [24] zu finden.

Für heutiges Intra-AS-Routing hat sich im Vergleich zu RIP das *Open Shortest Path First* (OSPF) [6] durchgesetzt, es gehört zur Klasse der *Link-State*-Protokolle. Unter den Routern werden dabei der Status sowie die Kosten einzelner Links zu Nachbarroutern periodisch signalisiert. Diese Signalisierungsdaten werden als *Link State Advertisement* (LSA) bezeichnet. Ihre Signalisierung erlaubt es jedem Router eine lokale Datenbank über die Links des Netzwerks aufzubauen und daraus einen gerichteten Graphen abzuleiten, der das Netzwerk und seine möglichen Pfade beschreibt. Basierend auf diesen Daten kann ein Router *Shortest-Path*-Berechnungen mit Hilfe des *Dijkstra*-Algorithmus ausführen und dadurch schleifenfreie Routen zu beliebigen Zielen im Netzwerk bestimmen.

OSPF Designated Router

Bei Verwendung von *n* gleichberechtigten Routern werden Signalisierungen insgesamt über $\frac{n*(n-1)}{2}$ Kommunikationsbeziehungen versandt, was einer quadratischen Komplexität von $O(n^2)$ entspricht. Zur Reduktion dieses Signalisierungsaufwands verwendet OSPF einen sogenannten *Designated Router* (DR), er übernimmt die zentrale Rolle als Verteiler von Routingdaten im Netzwerk. Andere Router senden ihm ihre LSA-Daten bezüglich lokaler Nachbarschaftsbeziehungen zu und erhalten auch von ihm die LSA-Daten fremder Router. Dadurch wird die Signalisierungskomplexität zu einer linearen von $O(n)$ reduziert, was ebenfalls die resultierende Ressourcenbelastung im Netzwerk geringer ausfallen lässt. Parallel zum *Designated Router* existiert stets ein *Backup Designated Router* (BDR), dieser übernimmt sobald der jeweilige DR ausfällt. Sowohl der DR als auch der BDR werden mit Hilfe einer Priorität ausgewählt. Diese wird manuell vom Netzwerkoperator pro Netzwerkschnittstelle eines Routers festgelegt. Nur der Router mit der höchsten Priorität übernimmt die zentrale Rolle zur Verteilung von Routingdaten im Netzwerk.

OSPF Areas

Bei Verwendung von OSPF besteht die Möglichkeit, ein AS in sogenannte *Areas* zu unterteilen, welche jeweils eine eigene Verwaltungszone darstellen. Jede besitzt ihren separaten DR (und BDR), welcher für die internen Signalisierungen und somit auch für die Verteilung von Routingdaten innerhalb des Netzwerkabschnittes zuständig ist. Zusätzlich unterstützt OSPF eine zweistufige Hierarchie für die erstellten *Areas*, welche durch Konfiguration einer sogenannten *Backbone Area* mit der *ID 0* festgelegt wird und zusätzliche Datenreduktion bei der Verteilung von Routingdaten ermöglicht. Sie muss dafür innerhalb des AS einzigartig sein und aus einer zusammenhängenden Topologie bestehen, sodass sie eine direkte Verbindung zu allen untergeordneten *Areas* besitzt. Für ein Routing zwischen den einzelnen untergeordneten *Areas* wird eine *Topologieaggregation* verwendet, wodurch die interne Struktur des jeweiligen Netzabschnitts gegenüber Nachbar-*Areas* abstrahiert wird. Ihnen wird ausschließlich eine sogenannte *aggregierte Route* übertragen, welche für alle internen Knoten gilt und deren gemeinsame Netzwerkadresse als Zielbeschreibung enthält. Dieser Vorgang wird im Folgenden als *Zielaggregation* bezeichnet und reduziert die Größe der übertragenen Signalisierungsdaten.

2.1.8.3 Inter-AS-Routing

Für ein Routing zwischen den verschiedenen AS wird im Internet das *Border Gateway Protocol* (BGP) [42] eingesetzt. Es ist ein *Path-Vector*-Protokoll, welches ähnlich zu *Distance-Vector*-Protokollen arbeitet und unter den Routern periodisch Tabellen über die lokal bekannten Routen austauscht. Dabei wird, ähnlich zu OSPF, eine Zielaggregation verwendet, sodass jede ausgetauschte Route für das jeweilige Zielnetzwerk gilt. Empfängt ein Router diese Routingdaten von einem Nachbarrouter, kombiniert er diese mit seinen lokalen Daten und errechnet daraus wiederum neue Routen, welche er seinen Nachbarn übermittelt. Dieser Prozess wird kontinuierlich wiederholt, sodass nach und nach jeder BGR die

Routen zu entfernten AS erlernt. BGP unterscheidet sich jedoch grundsätzlich von bekannten *Distance-Vector*-Protokollen:

- BGP beachtet nicht nur die kürzeste Route zu einem Ziel, sondern es werden auch Alternativpfade berücksichtigt.
- Während der Weiterleitung von Routingdaten wird der AS-Pfad aufgezeichnet und von BGP als zusätzliches Attribut einer Route übertragen. Er beschreibt mit Hilfe von AS-Nummern den Pfad, den das jeweilige Routingdatum zum aktuellen Router durchquert hat[6]. Merkt ein Router, dass sein lokales AS bereits in dieser Liste vorhanden ist, ignoriert er die jeweils empfangene Signalisierung und vermeidet dadurch, dass eine sogenannte *Routingschleife* in Routingtabellen und somit auch während der Paketweiterleitung entsteht. Zusätzlich wird der dadurch empfangen AS-Pfad zum sendenden Netzwerkknoten als Attribut für die jeweils signalisierte Route lokal abgespeichert und steht als Eingabe für die Realisierung von Netzwerkrichtlinien zur Verfügung, sodass beispielsweise bestimmte AS bei Routingentscheidungen explizit vermieden werden können.

Aufgrund der wirtschaftlichen Konkurrenz zwischen den einzelnen Netzwerkbetreibern von unterschiedlichen AS ist es vorteilhaft, dass die zweistufige Hierarchie der Routingprotokolle eine autonome Verwaltung des AS-internen Routings erlaubt. Dadurch kann ein Betreiber innerhalb seines AS entscheiden, für welchen seiner Kunden welche Art der Dienstleistung in Form der erbrachten Netzwerkkapazitäten bereitgestellt werden soll. Beim Einsatz von BGP können zusätzlich verschiedene Netzwerkrichtlinien zwischen den AS auf den BGR konfiguriert werden. Dadurch kann auch auf Inter-AS-Ebene sowohl die Weiterleitung von Routingdaten als auch von allgemeinen Datenpaketen beeinflusst werden. Als mögliche Kriterien eignen sich insbesondere die Ziel- oder Quellangabe sowie der jeweilige AS-Pfad. Es können auf Basis dieser Eingabedaten beispielsweise gezielt Routingsignalisierungen ignoriert oder erlaubt werden oder Datenpakete zum selben Ziel entlang unterschiedlicher Pfade geroutet werden. Letzteres wird mit Hilfe des bei BGP verwendeten Regelwerks umgesetzt, welches verschiedene Kriterien mit unterschiedlichen Prioritäten beachtet. Das resultierende Routing auf Inter-AS-Ebene ist somit nicht mehr nur eine Entscheidung für den kürzesten Weg, sondern es werden neue Möglichkeiten zur dynamischen Pfadwahl unterstützt, sodass auch über AS-Grenzen hinweg Verträge zwischen den Betreibern geschlossen werden können. Weitere Details zu möglichen Netzwerkrichtlinien mit BGP sind beispielsweise in [43] zu finden.

Route Aggregation

Da BGP im Internet insbesondere unter einer stark steigenden Anzahl von Einträgen in den Routingtabellen leidet [44], ist die sogenannte *Route Aggregation* insbesondere für die Reduktion der weitergeleiteten Menge von Routen interessant. Mehrere empfangene Routen werden dabei zu einer Route zusammengefasst und das jeweils übergeordnete Supernetzwerk wird als neues Ziel verwendet. Dadurch werden jedoch auch die bereitgestellten Details über die Topologie gegenüber anderen Routern reduziert.

Route Reflector

Insofern ein AS mehr als einen BGP-Router besitzt, wird das *Interior Border Gateway Protocol* (IBGP) als besondere Form von BGP verwendet, um Routingdaten zwischen mehreren BGR eines AS auszutauschen. Dabei besitzen alle IBGP-Router eines AS eine Verbindung zueinander, sodass daraus eine quadratische Kommunikationskomplexität resultiert. Ähnlich zur Rolle der *Designated Router* von OSPF wird bei BGP ein sogenannter *Route Reflector* (RR) als zentraler Ankerpunkt der Signalisierung zur Reduktion des Kommunikationsaufwands eingesetzt [45]. Die umliegenden BGP-Router werden als seine *Route Reflector Clients* konfiguriert und tauschen nur mit ihm Routingdaten aus. Zur weiteren

[6] Siehe Abschnitt 8.6 in [22]

Verbesserung der Skalierbarkeit für große Netzwerke können mehrere IBGP-Router als RR eingesetzt werden. In diesem Fall muss jedoch zwischen ihnen jeweils eine Verbindung aufrechterhalten werden, sodass sich dadurch ebenfalls eine quadratische Kommunikationskomplexität ergibt.

Confederation

Zur weiteren Reduktion des Kommunikationsaufwands von IBGP-Routern kann *AS Confederation* [46] eingesetzt werden. Dabei werden für ein AS durch den Netzwerkbetreiber sogenannte *sub-AS* konfiguriert, wodurch die quadratische Kommunikationskomplexität nur noch für einen Teil der ursprünglichen Netzwerkgröße notwendig ist. Diese Vorgehensweise ist vergleichbar mit einer expliziten Unterteilung des Netzwerks in sogenannte Cluster.

2.1.8.4 Vergleich zwischen Intra-AS- und Inter-AS-Routing

Die nachfolgende Tabelle vergleicht OSPF und BGP anhand der für diese Arbeit wichtigsten Kriterien miteinander. Insbesondere steht die autonome Konfiguration im Fokus der Betrachtungen.

Eigenschaft	OSPF	BGP (inter-AS)
Einordnung der Signalisierungen im OSI-Modell	als Protokoll auf Schicht 4	als Protokoll oberhalb von Schicht 4, TCP wird als Transportprotokoll verwendet
Sicherung von Signalisierungen	ja (eigene Mechanismen)	ja (durch Mechanismen von TCP)
Clusterbildung	ja (manuell in *Areas*)	ja (manuell in sub-AS)
Hierarchiebildung	ja (manuell mit max. 2 Stufen)	ja (manuell mit max. 2 Stufen)
Adresszuweisung	Nein	nein
Zielaggregation	Ja	ja
Verteilung von Routingdaten	*Link-States*	Routingtabellen mit AS-Pfaden (sowie zusätzliche Attribute)
Metrik	Hop-Distanz	Länge des AS-Pfads, Hop-Distanz zum nächsten Router, Priorität[7]
Routingstrategie	*Shortest Path*	*Shortest Path*, Beachtung von Netzwerkrichtlinien
Routenberechnung	*Dijkstra*-Algorithmus (verteilte Routinginstanzen)	regelbasiert (verteilte Routinginstanzen)
Routingzeitpunkt	proaktives Routing	proaktives Routing
Erreichbare Routingziele	alle Knoten des AS	alle AS
Möglichkeit zur Vermeidung von Routingschleifen	Ja	ja (notw. Erweiterungen existieren)

Tabelle 2.8: Vergleich zwischen OSPF und BGP

Aus Tabelle 2.8 wird ersichtlich, dass OSPF besonders für Intra-AS-Routing geeignet ist. Es bietet eine schnelle Konvergenz bei Topologieänderungen. Diese treten innerhalb eines AS typischerweise häufiger als auf Inter-AS-Ebene auf. Im Gegensatz dazu bietet BGP den Vorteil, große Domains verwalten zu können. Bei BGP muss ein Router nicht die gesamte Netzwerktopologie speichern. Im Vergleich zu OSPF liegt das Ergebnis der Routenberechnungen aufgrund des verteilt ablaufenden *Bellman-Ford* Algorithmus nach Empfang der Routingdaten unmittelbar vor. Beide Protokolle haben einen hohen admi-

[7] Zur Priorisierung können die Werte für *Origin*, *Local-Pref* oder auch *Multi-Edit-Discriminator* eingesetzt werden, um bei mehreren möglichen Routen eine von ihnen auf dem jeweiligen BGP-Router zu favorisieren.

nistrativen Aufwand durch den Netzwerkoperator gemeinsam. Die Anhänge A.1 und A.2 zeigen Ausschnitte über die für OSPF respektive BGP vorhandenen Konfigurationsparameter, sodass ein Eindruck über den erforderlichen Administrationsaufwand für beide Lösungen vermittelt wird.

2.2 Qualitätsanforderungen für Übertragungen

Während sich Abschnitt 2.1 mit der allgemeinen Übertragung von Anwendungsdaten durch Netzwerke beschäftigte, werden nun sogenannte Qualitätsanforderungen von Anwendungen näher betrachtet. Sie definieren Merkmale für den erbrachten Dienst des Netzwerks während der Übertragung der Anwendungsdaten. Die dabei erbrachte Dienstqualität wird auch als *Quality of Service* (QoS) bezeichnet. Um den Forderungen der Anwendung zu entsprechen, müssen ihre Anforderungen bei der Ermittlung von Routingentscheidungen im Netzwerk berücksichtigt werden. Daraus resultiert das sogenannte QoS-Routing, welches verschiedenen Qualitätsanforderungen unterstützen kann (siehe Abs. 1.2.1 in [47]):

- **Datendurchsatz:** Es ist entscheidend, wie viele Bytes pro Sekunde übertragen werden.
- **Verzögerung:** Es ist die resultierende Gesamtverzögerung während der Übertragung wichtig.
- **Jitter:** Es ist wichtig, dass eine Übertragung möglichst geringe Laufzeitunterschiede aufweist.
- **Paketverluste:** Es sollen möglichst wenige Pakete falsch übertragen oder verworfen werden.

Jede dieser Anforderungen kann entweder als *hard* oder *soft* interpretiert werden. Im ersten Fall gilt die jeweilige Anforderung als exakte Ober- oder Untergrenze, welche keinesfalls über- oder unterschritten werden darf. Im zweiten Fall gilt die Anforderung als Richtwert, welcher möglichst für die jeweilige Übertragung eingehalten werden soll. Dabei sind Wahrscheinlichkeitsangaben möglich, welche die Einhaltung der Werte in Bezug auf die Gesamtübertragung näher spezifizieren.

2.2.1 Relevanz im Internet

Entsprechend der Topologieunterteilung aus Abschnitt 2.1.8.1 können Router in Abhängigkeit von ihrer Position im Netzwerk und Aufgabenschwerpunkte klassifiziert werden. Abschnitt 14.2 in [24] beschreibt folgende Aufteilung:

- ***Core Router*:** Diese Router verbinden die Netzwerke eines ISPs. Der primäre Fokus liegt bei ihrer Funktion vor allem auf Paketdurchsatz und Zuverlässigkeit des Netzwerks. Insbesondere ist hier die Zeit für eine einzelne Paketübertragung entscheidend. Pakete sollen möglichst schnell das Netzwerk wieder verlassen, sodass die Ressourcen für nachfolgende Daten zur Verfügung stehen und hohe Datenraten ermöglicht werden.
- ***Edge Router:*** Ein solcher Router stellt das Bindeglied zwischen einem ISP-Netzwerk und einem Kunden dar, dazu zählen Firmen- und Privatanschlüsse. Ein *Edge Router* verbindet heterogene Zugangstechniken (*Digital Subscriber Line* (DSL), *Kabelmodem, Universal Mobile Telecommunications System* (UMTS)) und Protokolle (PPP, PPPoE) mit dem homogenen Kernnetzwerk des Providers. Im Vordergrund steht dabei vor allem der Paketdurchsatz, da Kundenanschlüsse stetig ausgebaut werden und somit mehr Kapazität vom Netzwerk des Providers verlangt wird.
- ***Enterprise Router:*** Diese Router befinden sich in großen Firmen oder Universitäten und verbinden verschiedene größere Netzwerkteile miteinander. Im Vordergrund steht dabei die kostengünstige Internetanbindung von möglichst vielen Endknoten. Zusätzlich spielt die Priorisierung von Kommunikationsströmen im Netzwerk eine Rolle. Daten spezieller Anwendungen besitzen definierte Anforderungen an die Qualität der Übertragung, welche im Netzwerk beachtet werden müssen. Des Weiteren sollte der Aufwand zum Management dieser Router möglichst gering ausfallen, um die Wirtschaftlichkeit des jeweiligen Netzwerkbetreibers zu unterstützen. Autonome Selbstkonfiguration und automatische Reaktion auf Veränderungen sind besonders wichtig.

Aus der vorgestellten Routerunterteilung geht hervor, dass Qualitätsanforderungen insbesondere für die sogenannten *Enterprise Router* wichtig sind. Dennoch müssen Qualitätsanforderungen entlang der gesamten Route beachtet werden, um sie vollständig zu erfüllen. Eine einzige ungünstige Pfadauswahl kann andernfalls einen Link einbeziehen, welcher beispielweise nicht die geforderte Datenrate ermöglicht oder eine zu hohe zusätzliche Verzögerung verursacht.

2.2.2 Klassenbasierte und strombasierte Anforderungen

Anforderungen können klassenbasiert festgelegt werden, sodass ein Router jedem Paket in Abhängigkeit von seiner Klasse eine vordefinierte Strategie zur Weiterleitung zuordnet. Dieses Verhalten wird als *Differentiated Services* (*DiffServ*) [4] bezeichnet. Die vordefinierten Klassen und Weiterleitungsstrategien gelten dabei stets nur für einzelne *DiffServ*-Domänen. Eine solche Domäne beinhaltet dabei alle Router eines Netzwerkbetreibers, welche den gleichen Netzwerkrichtlinien folgen. Die Klassenzugehörigkeit eines Pakets wird durch die Felder *Differentiated Services Codepoint* (DSCP) in IPv4 und *Traffic Class* in IPv6 innerhalb des jeweiligen Paketkopfes signalisiert. Ein Router kann auf Basis dieser Daten beispielsweise Audiopakete mit möglichst hoher Priorität weiterleiten. Ihre Übertragung soll einer möglichst geringen Gesamtverzögerung unterliegen, während Pakete sonstiger Datenströme mit normaler Priorität verarbeitet werden. Entscheidend für diesen Wechsel zwischen verschiedenen Weiterleitungsstrategien ist die Klassifikation von Paketen. Sie geschieht typischerweise auf den Routern am Rand der *DiffServ*-Domäne. Diese analysieren den Inhalt der Metadaten[8] der verwendeten Protokolle und legen innerhalb des IP-Paketkopfes die jeweilige Klasse des Pakets fest. Dieser Wert wird von den nachfolgenden Routern ausgewertet und das Paket entsprechend der vordefinierten Strategie weitergeleitet. *DiffServ* verlagert folglich die komplexere Paketklassifizierung in die Router am Rand der *DiffServ*-Domäne, während die inneren Router die nachfolgende Paketweiterleitung entsprechend der bereits vorliegenden Klassifikation ausführen. Zu diesem Zweck wird ihnen die Klassifikation in Form von zusätzlichen Metadaten der Pakete mitgeteilt, sodass sie keine lokalen Statusdaten speichern müssen. Dadurch wird eine zu starke Ressourcenbelastung der *Core Router* verhindert.

Als Alternative zur Klassifikation auf Paketbasis können Anforderungen auch je Datenstrom definiert werden. In diesem Fall wird statt einer Priorisierung von Paketen eine feste Ressourcenreservierung für den jeweiligen Datenstrom vorgenommen, dieses Verfahren wird *Integrated Services* (IntServ) genannt. Durch Signalisierungen zwischen Quelle und Ziel entlang der gewählten Route werden auf jedem (Zwischen-)Router lokale Ressourcen dem Datenstrom fest zugeordnet. Jeder Router speichert diese Zuordnungen ab, sie bleiben bis zu ihrer expliziten Auflösung (durch Signalisierung oder Erreichen eines Timeouts) als lokale Statusdaten bestehen. In heutigen Netzwerken wird das *Resource Reservation Protocol* (RSVP) [48] oder *Next Steps in Signaling* (NSIS) [49] zur Signalisierung eingesetzt, beide Protokolle dienen sowohl zur Reservierung als auch zur Auflösung selbiger und können in IP-Netzwerken eingesetzt werden. Jedes nach dem Reservierungsvorgang eintreffende Paket wird durch den jeweiligen Router nach wichtigen Metadaten (der Vorgang wird auch als *Packet Inspection* bezeichnet) untersucht und mit den lokalen Statusdaten verglichen. Dadurch kann es seinem Datenstrom und der zugehörigen Reservierung zugeordnet werden, sodass die Erfüllung der durch die Anwendung gestellten Anforderungen sichergestellt wird. Durch dieses Verfahren kann beispielsweise für einen Videostrom eine gewünschte Mindestdatenrate sichergestellt werden und somit Wiedergabelücken auf Empfängerseite vermieden werden.

Allgemein betrachtet ermöglicht *IntServ* im Gegensatz zu *DiffServ* eine feinere Unterscheidung zwischen den Qualitätsanforderungen einzelner Anwendungen. Eine Anwendung erhält dabei ebenfalls eine explizite Antwort, ob die Reservierung erfolgreich war oder fehlgeschlagen ist. Des Weiteren wird bei Misserfolg gemeldet, welche Qualitätsanforderungen nicht erfüllt werden können. Diese Vorteile werden zum Preis von zusätzlich gespeicherten Statusdaten auf den Routern erzielt. Aus diesem Grund ist

[8] Siehe Abschnitt 2.1.3

der Einsatz von *IntServ* auf *Core Routern* eher kritisch zu sehen und wird bei heutigen ISPs vermieden. Stattdessen unterstützen einige ISPs *DiffServ* auf ihren Routern. Weitere Details zur Anwendung von *DiffServ* und *IntServ* im heutigen Internet sind in [50] zu finden. Die Autoren diskutieren die notwendigen Mechanismen für die Einhaltung von Qualitätsanforderungen für eine Ende-zu-Ende Übertragung im Internet. Des Weiteren werden in [51] und [52] hybride Frameworks vorgestellt. Sie ermöglichen *IntServ* auf Basis von *DiffServ* für eine Ende-zu-Ende Übertragung unter Berücksichtigung von Qualitätsanforderungen. Im Vordergrund steht dabei eine Abbildung von *IntServ*-spezifischen Anforderungen auf das *DiffServ*-Modell. Etwaige Routingaspekte oder neuartige Routingalgorithmen werden dabei nicht erläutert. Eine ähnliche Arbeit stellt [53] dar. Ein weiterer hybrider Ansatz ist mit dem Ressourcenmanagementsystem *Darwin* [54] verfügbar. Dabei wird nicht nur die Reservierung von Linkkapazitäten betrachtet, stattdessen wird ein Framework für die Verwaltung von jeglichen Hardwareressourcen vorgestellt. Es kann somit ebenfalls zur Zuteilung von Prozessorzeiten oder Speicherkapazitäten verwendet werden, Routing steht dabei nicht im Vordergrund.

2.2.3 Audiovisuelle Datenströme

Sowohl von Privatpersonen als auch Großunternehmen werden audiovisuelle Datenströme für vielfältige Anwendungen eingesetzt. Dabei wird eine definierte Qualität für die Übertragung der Daten gefordert. Besonders hohe Anforderungen gelten für Videokonferenzen, was durch den Echtzeitcharakter der übertragenen Audio- und Videodaten begründet ist. Ihre Präsentation muss auf Empfängerseite mit möglichst geringer Verzögerung und in unveränderter Form möglich sein, sodass den Teilnehmern der Konferenz eine gute Interaktivität geboten wird.

2.2.3.1 Charakterisierung der Pakete

Audiovisuelle Daten werden typischerweise in komprimierter Form durch ein Netzwerk übertragen, um die notwendige Datenrate und somit auch die verbrauchten Ressourcen im Netzwerk möglichst gering zu halten. Dazu wird ein Audio- bzw. Videoencoder eingesetzt, der entsprechend der gewünschten Rate sogenannte Frames generiert. Für Audioströme beinhaltet ein solches Frame die Daten der Quelle für eine definierte Aufnahmezeit.

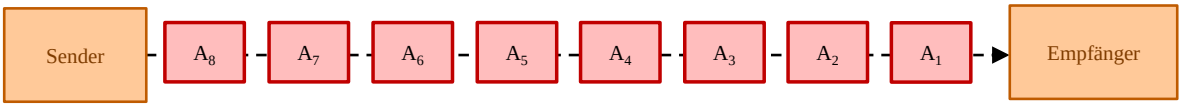

Abbildung 2.5: Aufbau eines Audiostroms

Zur Übertragung über das Netzwerk werden die Audioframes auf Pakete aufgeteilt und diese als eigenständige Einheiten durch das Netzwerk mit gleicher Wichtigkeit verschickt[9]. In Abbildung 2.5 ist ein Beispiel einer solchen Übertragung zu sehen, worin jedes Paket jeweils einem Audioframe entspricht.

Abbildung 2.6: Aufbau eines Videostroms

Bei Videoströmen enthält jedes Frame genau ein aufgenommenes Bild. Im Gegensatz zu Audioströmen werden dabei Frames mit unterschiedlicher Größe und Wichtigkeit verwendet. Ein Beispiel dafür ist in Abbildung 2.6 zu sehen. Der Sender generiert zu äquidistanten Zeiten entweder ein Vollbild, ein sogenanntes *key frame*, oder ein Differenzbild [55]. Während ersteres die Informationen jedes Bildpunktes

[9] Weitere Betrachtungen zur Häufigkeit, Dauer und zur benötigten Bandbreite von Audioframes sind beispielsweise in Kapitel 3 von http://www.telefonbau-schneider.de/uploads/media/VAF-Studie_Bandbreitenberechnung_in_VoIP-Netzen.pdf zu finden.

beinhaltet, sind in Differenzbildern ausschließlich Unterschiede zu Bildern der Vergangenheit bzw. Zukunft[10] gespeichert. Typischerweise ist ihre Größe gegenüber *key frames* wesentlich geringer. Erst durch die Kombination von Voll- und Differenzbildern ergibt sich während der Wiedergabe auf Empfängerseite die gewünschte Bildrate des Videos.

2.2.3.2 Paketverluste

Paketverluste können durch Überlastsituationen im Netzwerk jederzeit verursacht werden. Bei Audioströmen führt jedes verlorene Paket zu einem sehr kurzen Aussetzer der Wiedergabe. Insofern dabei nur vereinzelte Fragmente verloren gehen, stört dies die Verständlichkeit nicht signifikant. Je länger jedoch solche Phasen andauern, desto unverständlicher wird die Tonwiedergabe auf Empfängerseite.

Abbildung 2.7: Videoübertragung mit (links) und ohne (rechts) Paketverluste

Kritischer ist ein Paketverlust für Videoströme [56]. Insbesondere bei *key frames* bedeutet der Verlust eines Pakets nicht nur den Verlust des jeweiligen Vollbildes, sondern es geht für die Empfängerseite auch eine besonders wichtige Datenreferenz verloren. Als Folge daraus kann der Verlust eines Pakets ebenfalls eine signifikant falsche Wiedergabe nachfolgender Zwischenbilder verursachen, da der Empfänger die letzte vollständige Aktualisierung des Videobildes nicht korrekt wiedergeben kann. In Abbildung 2.7 sind die Auswirkungen von Paketverlusten zu erkennen. Die Aufnahme wurde mit der in Kapitel 5 vorgestellten Software *Homer-Conferencing* erstellt und vergleicht zwei gleichzeitig übertragene Videoströme mit unterschiedlicher Übertragungsqualität. Während im linken Bild der Kopf des dargestellten Hasen scheinbar noch an einer alten Position zu verbleiben scheint, entstehen im Bereich des Schmetterlings Bildartefakte mit falschen Farbwerten. Je nach Häufigkeit der Paketverluste kann sich eine solche Videowiedergabe auf Empfängerseite bis zur Unkenntlichkeit verschlechtern. Die Anwendung kann in diesem Fall durch Reduktion des jeweiligen Datenstroms reagieren, um die Auslastung der verwendeten Links im Netzwerk zu reduzieren und somit eventuellen Paketverlusten vorzubeugen. Ein erster Schritt kann dabei die Reduktion der Auflösung des übertragenen Videobildes darstellen. Dies führt gegenüber einer Reduktion der Bildwiederholrate zu einer eher akzeptablen Qualitätsverminderung auf Empfängerseite [57]. Sollte dies nicht genügen, kann die Bildwiederholrate reduziert oder ein vollständiger Wechsel des verwendeten Codecs durchgeführt werden.

2.2.3.3 Übertragungsverzögerungen

Sollten Links eine hohe Auslastung besitzen, kann dies die resultierende Ende-zu-Ende-Verzögerung zwischen Sender und Empfänger signifikant erhöhen[11]. Bei Übertragung eines gespeicherten Videoangebotes kann die Empfängerseite dies durch eine initiale Vorpufferung kompensieren. Beispielsweise dauert die Wiedergabe bei Zugriff auf ein Video des Anbieters *Youtube* anfänglich einige Sekunden und startet erst verzögert. Detailliertere Betrachtungen zur Pufferung sind in [58] zu finden. Im Gegensatz dazu sind bei Echtzeitanwendungen, beispielsweise bei Verwendung von Homer-Conferencing [11] o-

[10] In diesem Fall geschieht die Übertragung des Datenstroms verzögert um eine definierte Anzahl von Frames.
[11] Siehe Abschnitt 7.1.4 in [22]

der Skype, die auftretenden Übertragungsverzögerungen wesentlich kritischer in Hinblick auf die erreichte Dienstqualität der Anwendung zu sehen. In [59] wird für diese Übertragungen audiovisueller Datenströme durch die *International Telecommunication Union* (ITU) eine Obergrenze von 400 ms für die Ende-zu-Ende-Verzögerung beschrieben. Für höhere Verzögerungen wird die Qualität einer Audioübertragung als überwiegend inakzeptabel eingeschätzt. Des Weiteren wird ein Wert von etwa 150 ms genannt, ab dem die Zufriedenheit von Nutzern zu sinken beginnt[12]. Als Gegenstück zu diesen Schwellwerten gibt es allgemeine Messungen über die in Netzwerken auftretenden Verzögerungen in Abhängigkeit von der Tageszeit. Das Internet stellt dabei als größtes Netzwerk eine besondere Herausforderung für die Einhaltung von Verzögerungszeiten dar. Die Ergebnisse des *Test Traffic Measurement Service* Projektes [60] des *RIPE Network Coordination Centre* (RIPE NCC)[13], welche in [61] ausgewertet wurden, zeigen beispielsweise, dass die für ausgewählte Routen des Internets gemessenen durchschnittlichen Verzögerungszeiten deutlich unter 400 ms liegen. In [59] wird durch die ITU die Verzögerung für viele innerkontinentalen Übertragungen (Afrika, Europa, Nordamerika mit Übertragungsdistanzen von weniger als 5000 km) sogar auf Werte kleiner 150 ms eingeschätzt. Ausnahmen können dabei exotische Netzwerkstrukturen sein, welche beispielsweise Satellitenlinks beinhalten.

2.2.3.4 Anforderungen an den Routingalgorithmus

Für eine möglichst hohe Übertragungsqualität müssen die Datenpakete einer Anwendung entsprechend folgender Ziele durch das Netzwerk übertragen werden:

- **Vermeidung von überlasteten Links:** Überlastete Hardware verursacht Paketverluste, welche durch gezielte Nutzung alternativer Routen vermieden werden können [62].
- **Vermeidung von Links mit ungenügenden Eigenschaften:** Links mit unzureichenden physikalischen Eigenschaften in Abhängigkeit von Qualitätsanforderungen der Anwendung sollten generell vermieden werden. Beispielsweise sollten Links mit zu geringer Übertragungsgeschwindigkeit oder ungewöhnlich hoher Verzögerung für die Übertragung audiovisueller Echtzeitdaten vermieden werden.

Durch eine adaptive Pfadauswahl in Abhängigkeit von den verfügbaren Kapazitäten im Netzwerk können diese Ziele erreicht werden. Dazu müssen folgende Werte jedes Links durch den Routingalgorithmus beachtet werden:

- **Verfügbare Datenrate:** Die möglichen Datenraten entlang der vorhandenen Routen müssen in die Entscheidung einbezogen werden. Dadurch können Routen mit unzureichenden Kapazitäten vermieden und, insofern sie existieren, entsprechende Alternativrouten ausgewählt werden. Des Weiteren kann durch Auswertung verfügbarer Datenrate von Links automatisch der Pfad mit den meisten verbleibenden Ressourcen gewählt werden, um eine Lastverteilung im Netzwerk zu unterstützen.
- **Zu erwartende Verzögerung:** Im Netzwerk können Satellitenlinks existieren, welche typischerweise hohe Verzögerungen verursachen. Diese sollten für die Daten von Videokonferenzsitzungen im Rahmen der Möglichkeiten der Netztopologie vermieden werden. Zu diesem Zweck sollte bei einem QoS-Routing die minimal zu erwartende Verzögerung von Links in die Routingentscheidung einbezogen werden.

[12] Weitere Details zu dem dabei verwendeten *E-model* sind in ITU-T G.107 zu finden.
[13] Die RIPE NCC ist in Europa zuständig für die zentrale Vergabe von IP-Adressbereichen sowie AS-Nummern. Die durch die RIPE durchgeführten Messungen fokussieren auf den Verzögerungen zwischen Knoten in Europa, USA und Neuseeland.

2.2.4 Routingstrategien

In Abschnitt 2.1.6.1 wurden die beiden Routingstrategien *Shortest Path* und *Widest Path* beschrieben. Beide können für ein QoS-Routing kombiniert werden. Entscheidend ist dabei die Priorisierung der unterschiedlichen Zielstellungen. Folgende Routingstrategien können daraus abgeleitet werden:

- **Shortest Widest Path First (SWPF)** [63]**:** In diesem Fall wird stets der Pfad mit den geringsten Kosten bewertet, welcher im Vergleich zu Alternativrouten die besten verfügbaren Eigenschaften besitzt. Zur Bewertung dient die je Route verfügbare Datenrate, zusätzlich kann die zu erwartete Verzögerung einbezogen werden. Als weiteres Kriterium werden die Routenlängen verwendet.
- **Widest Shortest Path First (WSPF)** [8]**:** Im Gegensatz zu SWPF werden die geringsten Kosten der Route zugeordnet, welche die kürzeste Routenlänge aufweist. Existieren mehrere solcher Routen, wird zusätzlich die Datenrate als sekundäres Kriterium hinzugenommen, eine höhere Datenrate führt dabei zu geringeren Kosten. Dieser Ansatz ähnelt sehr dem reinen BE-Routing, da beide die Routenlänge als primäres Entscheidungskriterium verwenden.
- **Best Fit First (BFF):** Im Gegensatz zu den beiden vorherigen Strategien wird stets die Route verwendet, die möglichst genau die geforderten Eigenschaften (verfügbare Datenrate und erwartete Verzögerung) besitzt. Dadurch werden Routen mit besseren Eigenschaften für Übertragungen mit stärkeren Anforderungen zurückgehalten.

Unabhängig von der gewählten Strategie für ein Routing unter Beachtung von Qualitätsanforderungen müssen die Kapazitäten existierender Routen bekannt sein. Sie müssen durch das jeweils verwendete Routingprotokoll unter den beteiligten Routern signalisiert werden, sodass sie dem Routingalgorithmus zur Verfügung stehen.

2.2.5 Aggregation von Netzwerkpfaden

Ähnlich der in Abschnitt 2.1.8 beschrieben Zielaggregation wird für ein QoS-Routing häufig eine zusätzliche Topologieaggregation für die physikalischen vorhandenen Pfade innerhalb eines Netzwerkabschnittes unter Einbeziehung der QoS-Eigenschaften eingesetzt. Dies dient der Datenreduktion während der Signalisierung von Routingdaten durch ein Routingprotokoll. Dabei werden interne Netzwerkstrukturen mit Hilfe von aggregierten Routen beschrieben und zwischen den beteiligten Routern signalisiert. Die Anzahl der resultierenden Routen sollte dabei stets geringer als die Anzahl von physikalisch vorhandenen Pfaden sein. Dennoch kann die Menge der physikalisch existierenden Pfade durch das Netzwerk entweder durch eine oder auch mehrere aggregierte Routen abgebildet werden. Letzteres spielt insbesondere bei konkurrierenden QoS-Eigenschaften eine Rolle, welche mit Hilfe einer multi-dimensionalen Aggregation detailliert abgebildet werden können. In diesem Fall wird jede aggregierte Route mit einer individuellen Priorisierung zwischen den möglichen QoS-Eigenschaften gebildet. Routenberechnung können dadurch beispielsweise mit Fokus auf die Datenrate oder die Verzögerung durchgeführt werden, da für beide Varianten unterschiedliche Topologieabbildungen (aggregierte Routen) zur Verfügung stehen. Des Weiteren kann der Detailgrad der Abbildung zusätzlich durch sogenannte logische Knoten gesteigert werden. Sie besitzen keine physikalischen Entsprechungen, sondern sie repräsentieren ausgewählte Bereiche der physikalisch vorhandenen Topologie. Eine aggregierte Route zu einem solchen Knoten beschreibt Pfade zu allen Knoten des repräsentierten Netzwerkabschnittes.

Im Allgemeinen bestimmt die Aggregationsstrategie die Anzahl der resultierenden aggregierten Routen und somit auch die resultierende Genauigkeit der Abbildung der physikalisch vorhandenen Netzwerktopologie. Innerhalb dieser Arbeit spielen die multi-dimensionalen Aggregationen, welche für jede QoS-Eigenschaft eine unabhängige Aggregation der physikalisch vorhandenen Netzwerktopologie beinhal-

ten, eine untergeordnete Rolle und werden in diesem Abschnitt nicht näher betrachtet. Stattdessen werden die Möglichkeiten zur einfachen Aggregation näher erläutert, wobei die Datenrate gegenüber der Verzögerung höher priorisiert wird.

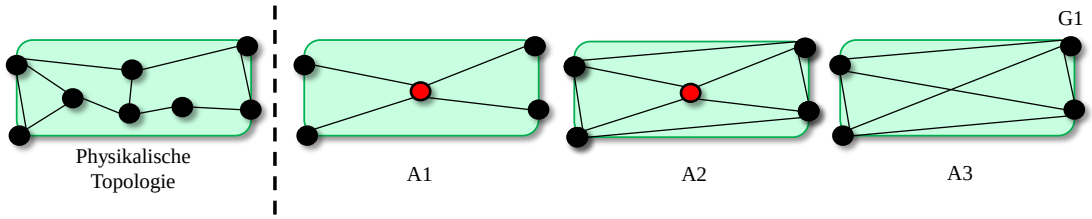

Abbildung 2.8: Ausgewählte Strategien zur Aggregation von physikalisch vorhandener Topologie

Abbildung 2.8 stellt die physikalisch vorhandene Topologie eines Netzwerkabschnittes auf der linken Seite drei ausgewählten Aggregationsvarianten der Literatur [64] auf der rechten Seite gegenüber:

1.) **Sterntopologie ohne Bypass (A1):** Diese Strategie bildet aggregierte Routen zwischen allen Randroutern des Netzwerkabschnittes und einem zentralen, logischen Knoten (in Abbildung 2.8 rot markiert). Dieser besitzt keine physikalische Entsprechung, er steht stellvertretend für alle internen Knoten des Abschnittes. Die resultierenden aggregierten Routen können sowohl für ein Routing zu internen Knoten als auch zur Durchquerung des Netzwerkabschnittes verwendet werden. Jedoch ist dabei der Detailgrad der verbleibenden Topologiebeschreibung reduziert.
2.) **Sterntopologie mit Bypass (A2):** Diese Strategie ist eine Erweiterung von Variante A1, dabei werden zusätzliche *Bypass*-Pfade zwischen ausgewählten Randroutern verwendet. Sie bieten insbesondere bei der Durchquerung des Netzwerkabschnittes einen erhöhten Detailgrad für notwendige Routenberechnungen. Im Gegensatz zu A1 erhöht sich dadurch ebenfalls die Menge der zu signalisierenden Routingdaten.
3.) **Maschentopologie zwischen Gateways (A3):** Die dritte vorgestellte Strategie verwendet ausschließlich aggregierte Routen zwischen allen Randroutern des jeweiligen Netzwerkabschnittes. Durch wird ein guter Detailgrad bei der Routenberechnung zur Durchquerung des Netzwerkabschnittes realisiert.

Die resultierende Wahl der Topologieaggregation ist stets ein Kompromiss aus Datenreduktion und verbleibender Genauigkeit der Abbildung von physikalisch vorhandener Topologie. Je detaillierter die Abbildung die Realität wiedergibt, desto eher können suboptimale Routingentscheidungen verhindert werden.

2.2.6 Erweiterungen für heutige Routingprotokolle

Sowohl für das Routingprotokoll OSPF als auch für BGP existieren Konzepte zur Erweiterung, sodass zusätzlich Informationen zur Beschreibung der Kapazitäten von Routen unter den beteiligten Routern signalisiert werden. Diese werden nachfolgend als typische Lösung für IP-Netzwerke näher beschrieben. Zusätzlich wird ein Überblick über das *Private Network-Network Interface* (PNNI) gegeben, welches für ATM-basierte Netzwerke eingesetzt wird.

2.2.6.1 Intra-AS-Routing

Für OSPF existiert eine experimentelle Protokollerweiterung [8], sie legt für ein QoS-Routing folgende Aspekte fest:

- **Signalisierung von Routingdaten:** Im Gegensatz zu OSPF werden zusätzlich die Datenrate und die Verzögerung zur Beschreibung von Links verwendet. Diese werden innerhalb der von OSPF bekannten *Link State Advertisements* unter den Routern signalisiert.

- **Routingstrategie:** Des Weiteren legt die Spezifikation die Routingstrategie fest, wobei prinzipiell zwischen proaktiven und reaktiven Routenberechnungen unterschieden wird. Im ersten Fall wird *Widest Shortest Path First (WSPF) Routing* unter Verwendung des Bellman-Ford-Routingalgorithmus angewandt, um stets Routen mit der kürzesten Weglänge zu verwenden, bei gleichen Routenlängen wird als zusätzliches Kriterium die jeweils verfügbare Datenrate der Routen verwendet. Im zweiten Falle (reaktives Routing) kommt *WSPF-Routing* auf Basis des *Dijkstra*-Routingalgorithmus zum Einsatz. Für die Protokollerweiterung ist nicht exakt spezifiziert, wann welche der beiden Methoden zum Einsatz kommt. Stattdessen werden durch den Autor eher proaktive Routenberechnungen favorisiert. Diese werden periodisch durchgeführt, sodass die lokalen Routingtabellen aktuell gehalten werden.

Aus Abschnitt 3.5 in [8] lassen sich die offenen verbleibenden Punkte der Protokollerweiterung entnehmen:

- **Spezielle *Link-State-Advertisement*-Nachrichten** Nicht jeder in OSPF genutzte Typ von *Link State Advertisements* wird beachtet. Stattdessen werden QoS-Aspekte speziell für das Routing innerhalb einer OSPF *Area* betrachtet und auf das QoS-Routing zwischen OSPF *Areas* wird nicht näher eingegangen.
- **Interoperabilität:** Existieren innerhalb einer OSPF *Area* auch Router ohne QoS-Unterstützung, ist durch die Spezifikation keine explizite Reaktion festgelegt.

Die Protokollerweiterung kann trotz der oben genannten Punkte für ein QoS-Routing innerhalb einer OSPF *Area* verwendet werden. Dabei ist sowohl ein Einsatz für *DiffServ* als auch *IntServ* möglich. Ähnlich der originären OSPF-Spezifikation beruht die Erweiterung auf Annahmen bezüglich einer manuell festgelegten Netzwerk- und Adressierungsstruktur, das beinhaltet die Netzunterteilung in *OSPF Areas* als auch die damit verbundene notwendige Verteilung von IP-Adressen.

Ähnlich der vorgestellten Erweiterung für OSPF bietet *OSPF Traffic Engineering* (OSPF-TE) [65] eine Spezifikation zur Verteilung von erweiterten Routingdaten innerhalb von OSPF-Signalisierungsnachrichten. Innerhalb der Spezifikation wird der Einsatz für intra-*Area* Netzwerkabschnitte beschrieben. Die verwendeten Annahmen basieren ebenfalls auf denen von OSPF.

Als alternative Erweiterung für OSPF ist QOSPF [7] zu nennen, es kann sowohl für intra-*Area* als auch inter-*Area*-Routing verwendet werden. Zu diesem Zweck verteilen beteiligte Router durch spezielle Nachrichten Informationen über die verfügbaren Eigenschaften von Links. Das Protokoll ist insbesondere für strombasierte Anforderungen und den damit verbundenen Ressourcenreservierungen gedacht. Eine einmal für einen Datenstrom bestimmte Route kann daher nicht ohne aufwendige Neusignalisierung des gesamten Reservierungsvorganges verändert werden. Des Weiteren können die durch QOSPF eingeführten Nachrichten und deren Signalisierung für große Netzwerke schnell zu einem hohen Datenaufkommen im Netzwerk führen. Insbesondere die spezifizierte Signalisierung von durchgeführten Reservierungen unter den Routern kann mit zunehmender Netzwerkgröße und der Anzahl von Datenströmen problematisch werden. Der Ansatz beruht ebenfalls auf den Annahmen von OSPF (bspw. manuelle Netzwerkunterteilung und vorab vergebene Adressen).

2.2.6.2 Inter-AS-Routing

Ein auf *IntServ* basierendes QoS-Routing ist für *Core Router* nicht zu empfehlen. Entsprechend Abschnitt 2.2.1 müssen *Core Router* sehr viele Datenströme in sehr kurzer Zeit verarbeiten können. Im Gegensatz dazu verlangt *IntServ*, dass jeder Router zusätzliche Statusdaten pro Datenstrom speichert und eintreffende Pakete den bekannten Datenströmen zuordnet. Dafür müssen die notwendigen Metadaten aus dem jeweiligen Paketkopf extrahiert und als Basis für die Identifikation des Datenstroms in-

nerhalb der lokalen Daten verwendet werden. Dies führt zu einem erhöhten Speicherverbrauch und verursacht höhere Laufzeiten während der Paketweiterleitung. Als Folge daraus verwendet man innerhalb der Kernstrukturen eines Providernetzwerks stattdessen *DiffServ*. Es ordnet Pakete auf den Routern am Rande des Netzwerks einer Qualitätsklasse zu und markiert dies entsprechend. Nachfolgende Router werten diese Markierungen aus und wählen eine entsprechende Strategie zur Weiterleitung des jeweiligen Pakets. Dadurch werden Datenströme gegenüber anderen priorisiert, sodass ihre Weiterleitung erwartungsgemäß bevorzugt wird.

Die Erweiterung *QoS Policy Propagation via BGP* (QPPB) ermöglicht es, mit Hilfe von BGP eingehende Pakete entsprechend vorgegebener Klassen zu markieren, sodass nachfolgende Router das gewünschte Weiterleitungsverhalten entsprechend der jeweiligen Klasse anwenden. Dadurch können beispielsweise Pakete eines AS mit Vorrang gegenüber Paketen anderer AS weitergeleitet werden. Eine weitere Möglichkeit ist die Ratenbegrenzung in Abhängigkeit von der jeweiligen Paketquelle. Durch QPPB wird eine Vielzahl von Netzwerkrichtlinien unterstützt, was einem ISP ermöglicht, Verträge mit unterschiedlich zugesicherter Übertragungsqualität zu vereinbaren. Zur Zuordnung der Pakete zur jeweiligen Netzwerkrichtlinie (Vertrag) werden typischerweise eines oder mehrere der folgenden Kriterien verwendet:

- **Quell- und Zieladresse der Pakete:** Wenn das jeweilige Paket von einem Premiumkunden stammt, der besondere Übertragungsqualität gebucht hat, sollten seine Datenströme mit hoher Priorität durch das Netzwerk verarbeitet werden.
  Wurde einem Kunden der Zugang zu einem Dienst insbesondere zugesichert, sollten Datenströme in Richtung des jeweiligen Dienstservers bevorzugt übertragen werden.
- **AS-Pfad der Pakete**: Datenströme können beispielsweise entlang einiger Routen nur mit begrenzter Datenrate übertragen werden.

Die Klassifikation erfolgt somit stets auf Basis der Metadaten von IPv4/v6 sowie existierender Routingdaten von BGP. Die Klassifizierungsregeln werden dabei für jede Netzwerkschnittstelle festgelegt, wodurch verschiedene Pfade zwischen Quelle und Ziel einzelner Datenströme unterschieden werden.

2.2.6.3 Private Network-Network Interface

Als weiterer relevanter Ansatz zur Umsetzung von QoS-Routing ist das *Private Network-Network Interface* (PNNI) [66] zu nennen. Im Gegensatz zu OSPF und BGP ist PNNI für Netzwerke unter Verwendung des *Asynchronous Transfer Mode* (ATM) [67] entwickelt wurden. ATM unterscheidet sich dabei grundlegend von IP, indem es statt IPv4/v6-Paketen von dynamischer Größe sogenannte *Cells* mit konstanter Größe von 53 Bytes verwendet. Dabei kommen eine eigene Adressierung und Protokollstackarchitektur während der Signalisierungen zwischen den ATM-Switches zum Einsatz. Das Netzwerk ist dabei anhand der jeweils verwendeten Adressierungsschemata der Switches in sogenannte *Peer Groups* (PG) unterteilt. Für jede *Peer Group* wird ein *Peer Group Leader* (PGL) gewählt, welcher die PG innerhalb der nächsthöheren *Peer Group* repräsentiert. Die Wahl des PGL wird dabei durch die vergebenen Prioritäten und ATM-Adressen beeinflusst. Durch wiederholte Anwendung dieses Systems entsteht eine Hierarchie, welche maximal 104 Levels beinhalten darf. Jeder PGL ist darin zuständig für den Austausch von Routingdaten auf Basis eines *Link-State*-Protokolls. Dabei wird eine Zielaggregationen zur Reduktion der Signalisierungs- und Speicherkomplexität angewandt. Ein PGL kommuniziert aggregierte Topologiebeschreibungen seiner PG zu anderen PGLs des gleichen Hierarchielevels [68]. Umgekehrt leitet er empfangene Topologieinformationen anderer PGLs des gleichen Hierarchielevels zu den untergeordneten Mitgliedern seiner PG. Dies beinhaltet beispielsweise die Kosten für eine Route durch die externe PG.

Bei PNNI kann die Quelle bereits die Gesamtroute zum Ziel auswählen, insofern sie alle notwendigen Routingdaten besitzt. Dabei kommt *Shortest Path* Routing unter Beachtung von Qualitätsanforderungen

zum Einsatz. Das Ergebnis wird in der *Designated Transit List* (DTL) innerhalb der verschickten Signalisierungspakete abgespeichert und steht somit für die nachfolgenden Netzwerkabschnitte zur Verfügung. Des Weiteren ist es möglich, dass ein Switch eine Teilroute bis zu einer entfernten Zwischenstation explizit bestimmt und innerhalb der DTL der verschickten Signalisierungspakete beschreibt. Sollte in diesem Fall die Zwischenstation keine weitere Route zum Ziel bestimmen können – das Routing schlägt fehl – wird der sogenannte *Crankback*-Ablauf mit *Alternate Path Routing* durchgeführt. Dies kann beispielweise auftreten, wenn Qualitätsanforderungen nicht mehr erfüllbar sind. In diesem Fall wird die Signalisierung an einen vorhergehenden Switch zurückdelegiert und das Routing an dieser Stelle fortgesetzt. Dabei wird die bisher verwendete Route vermieden und eine alternative Route bestimmt. Dieses Verfahren wurde in [69] ebenfalls auf MPLS und GMPLS RSVP-TE angewandt.

Ähnlich wie OSPF und BGP setzt PNNI eine bereits vorkonfigurierte Netzstrukturierung und eine damit verbundene Adresszuweisung voraus. PNNI gilt als sehr komplex und seine Anwendung wird im heutigen – hauptsächlich aus IP-basierten Teilnetzwerken bestehenden – Internet als besonders wartungsaufwendig eingestuft. Aufgrund der damit verbundenen hohen Kosten wird ATM, und damit auch PNNI, aus heutigen Netzwerken durch OSPF und BGP verdrängt[14]. Dennoch ist es aufgrund der hierarchischen Struktur und der Unterstützung von Qualitätsanforderungen für sehr große Netzwerke relevant für die vorliegende Arbeit.

2.3 Ausgewählte Forschungsarbeiten

Auf dem Gebiet des Routings unter Berücksichtigung von Qualitätsanforderungen wird bereits seit vielen Jahren intensiv Forschung betrieben. In der Literatur findet man diese vor allem unter den Begriffen *QoS Routing* und *Constraint Based Routing* vor. Insbesondere um das Jahr 1998 entstanden konzentriert Veröffentlichungen auf dem Gebiet des QoS-Routings, insbesondere für mobile Netzwerke. Dabei sind verschiedenste Konzeptionen für Routingprotokolle entstanden. Ein Überblick dazu ist in [70] für allgemeine Unicast-basierte Übertragungen, in [71] [72] [73] [74] [75] [76] für allgemeine mobile Netzwerke und in [77] [78] [79] für Sensornetzwerke zu finden. Des Weiteren werden mögliche QoS-Routingalgorithmen in [80] und [81] zusammengefasst. In [82] werden existierende Lösungen für ein QoS-Routing unter Benutzung von möglichst disjunkten Pfaden, sogenanntes *Multipath Routing*, diskutiert.

Wie in Abschnitt 2.1.6.2 erläutert, wird bei den existierenden Lösungen grundsätzlich zwischen quellbasiertem und verteiltem Routing unterschieden. Orthogonal dazu kann eine Hierarchie aus Managementinstanzen zum Einsatz kommen. Die folgenden Abschnitte beschreiben diese Ausprägungen auf Basis der Ausführungen von [83] näher. Jeder Abschnitt enthält ein ausgewähltes Beispiel, welches jeweils kurz charakterisiert wird.

2.3.1 Quellbasiertes QoS-Routing

Bei dem sogenannten quellbasierten QoS-Routing besitzt der Quellknoten der jeweiligen Übertragung ausreichend Routingdaten, um eine vollständige Route zum Ziel zu bestimmen. Dies geschieht typischerweise auf Basis eines lokal gespeicherten Graphen, der die Topologie des Netzwerks abbildet. Jeder Link des Graphen entspricht einer physikalischen Verbindung zwischen zwei Netzwerkknoten. Abhängig von den signalisierten Daten des Routingprotokolls sind dabei für jeden Link seine QoS-Eigenschaften bekannt. Da der Quellknoten die gesamte Route berechnet, muss er das Ergebnis seiner Routenberechnung im Kopf des jeweiligen Pakets für nachfolgende Router speichern, sodass es für die weitere Paketweiterleitung beachtet werden kann.

Im Gegensatz zu verteiltem Routing kennt bei quellbasiertem Routing die Quelle alle Details über die Topologie des Netzwerks, sodass sie Schleifen in der Paketweiterleitung vermeiden kann. Die für eine

[14] Siehe beispielsweise: http://www.teltarif.de/telekom-all-ip-analoganschluss/news/50087.html

Berechnung benötigten Signalisierungen sowie ihre Implementierung können im Vergleich zu verteiltem Routing eher einfach gehalten werden. Nachteilig ist jedoch die dabei durch das Netzwerk übertragene Menge an Routingdaten, welche die QoS-spezifischen Informationen über das Netzwerk auf jedem Knoten stets aktuell halten. Aufgrund der quadratischen Kommunikationskomplexität ist dies insbesondere im Fall von häufigen Änderungen von Ressourcenreservierungen für sehr große Netzwerke problematisch. Des Weiteren können Skalierungsprobleme durch die Zentralisierung der Berechnungen auftreten.

Quellbasiertes Routing ist beispielsweise in [63] beschrieben, eine Routenberechnung erfolgt dabei auf dem jeweiligen Sendeknoten unter Beachtung der Bandbreiten und Verzögerungen entlang einzelner Links. Zwischen beiden QoS-eigenschaften wird eine Priorisierung verwendet, sodass der Fokus auf einer möglichst guten Verzögerung liegt und Links mit zu geringer verfügbarer Bandbreite ignoriert werden.

2.3.2 Verteiltes QoS-Routing

Wie in Abschnitt 2.1.6.2 beschrieben, wird bei einem verteilten Routing der resultierende Pfad durch die Entscheidungen von Zwischenknoten festgelegt. Erst die Kombination dieser Einzelentscheidungen ergibt das Gesamtrouting. Dafür muss ein Router mindestens den ausgewählten Teil der Netzwerktopologie kennen, den er für diese Routingentscheidung benötigt. Im Vergleich zu quellbasiertem Routing ist ein verteiltes Routing aufgrund der Verteilung der Speicher- und Berechnungslast skalierbarer. Dabei können jedoch Inkonsistenzen zwischen den Graphen einzelner Knoten auftreten, sodass Routingschleifen verursacht werden können. Werden die verteilten Routingentscheidungen aber auf Basis globaler Datengraphen berechnet, ergeben sich ebenfalls die bekannten Probleme von quellbasiertem Routing.

Beispielsweise wird in [84] für ein verteiltes QoS-Routing ein zweistufiger Verbindungsaufbau auf Basis von sogenannten *Selective Probes* vorgestellt. Der Ansatz geht von vorhandenen Routingdaten über die Topologie des Netzwerks aus, wobei nicht näher spezifiziert wird, ob diese über ein *Link-State-*, *Distance-Vector-* oder *Path-Vector*-basiertes Signalisierungsprotokoll im Netzwerk verteilt werden. Der Verbindungsaufbau einer Anwendung läuft wie folgt ab:

- **Phase 1:** Für eine Routinganfrage wird das sogenannte *Selective Probing* eingesetzt. Dabei werden *Selective-Probe*-Nachrichten vom Quellknoten verschickt. Diese stellen Routinganfragen dar und breiten sich konzentrisch vom Quellknoten entlang der existierenden Links und Knoten im Netzwerk aus. Es werden nur Links verwendet, die den Qualitätsanforderungen der Routinganfrage genügen. Das Verfahren verwendet zwei Durchläufe. Im ersten Schritt werden *Probe*-Nachrichten entlang des jeweils kürzesten Pfades versendet. Wird dadurch keine passende Route gefunden, werden im zweiten Durchlauf mit neuen *Probe*-Nachrichten alle alternativen Links mit steigender Routenlänge auf ihre Einsetzbarkeit geprüft. Wird eine *Probe*-Anfrage vom Zielknoten empfangen, ist eine passende Route gefunden und es wird keine weitere *Probe*-Nachricht versendet.
- **Phase 2:** Unter den gefunden Lösungen wählt der Zielknoten die Beste aus und schickt entlang dieser Route eine Bestätigung zum Quellknoten. Auf jedem Zwischenknoten werden dadurch Ressourcen für die jeweilige Verbindung reserviert. Sobald die Bestätigung am Quellknoten eintrifft, ist die Verbindung vollständig.

Sollten zwischen Phase 1 und 2 Veränderungen in der Topologie oder an Linkeigenschaften aufgetreten sein, ist es möglich, dass die Bestätigung nicht erfolgreich ist. In diesem Fall wird die Bestätigung abgebrochen und den vorhergehenden Routern der Abbruch signalisiert. Die bereits reservierten Ressourcen im Netzwerk werden daraufhin wieder freigeben.

Als Erweiterung der *Probe*-Anfragen werden in [85] von den gleichen Autoren sogenannte Tickets eingeführt. Ihre Anzahl wird für eine Routinganfrage automatisch vom System begrenzt vergeben, um die Menge der konzentrisch weitergeleiteten *Probe*-Signalisierungen im Netzwerk zu reduzieren.

2.3.3 Hierarchien

Um Skalierungsprobleme zu vermeiden, wird in Routingprotokollen häufig eine Hierarchie eingesetzt. Die Grundidee bei der Anwendung von Hierarchien besteht darin, durch geeignete Ebenen der Aggregation eine ausgewogene Verteilung von Speicher- und Berechnungslast im Netzwerk sicherzustellen. Die Knoten besitzen dadurch nur noch einen partiellen Graphen über die vorhandene Topologie im Netzwerk. Zusätzlich werden darin Topologiedetails abstrahiert, wobei einzelne Knoten, im Folgenden als logische Knoten bezeichnet, ganze Netzwerkabschnitte repräsentieren können. Für das Routing bedeutet dies eine Zielaggregation, wie sie in Abschnitt 2.1.8 beschrieben ist. Der Nachteil der Aggregation ist ein Verlust von Detailschärfe [86]. Bei jedem Abstraktionsschritt werden detaillierte Topologieinformationen zusammengefasst und nachfolgend als Routingdaten verwendet. Insbesondere bei QoS-Routing kann dies dazu führen, dass der Routingalgorithmus existierende Routen nicht kennt. Er meldet einen Routingfehler, obwohl womöglich eine Route für die geforderten Qualitätsanforderungen existiert. Alternativ kann der Fall eintreten, dass eine schlechtere Route einer besseren vorgezogen wird. Daraus ergibt sich der sogenannte *stretch factor*, der die mögliche Verschlechterung des Routings bei Verwendung einer Hierarchie beschreibt. Weitere Details dazu sind in [87] [88] zu finden.

Trotz der genannten möglichen Beeinträchtigungen der Routingperformanz wurden Hierarchien schon in der Vergangenheit angewandt und kommen auch in der Gegenwart zum Einsatz. Hierarchien versprechen ein gutes Skalierungsverhalten für große Netzwerke. Die ersten Arbeiten zu hierarchischem Routing begannen vor etwa 40 Jahren mit [89] und wurden in [88] verbessert. Als eine der ersten hierarchischen, hybriden Lösungen mit Verwendung von proaktivem und reaktivem Routing wurde in [90] das Konzept des *Zone Routing* Protokolls (ZRP) vorgestellt. Es unterteilt das Netzwerk in Zonen und nutzt innerhalb einer Zone stetig aktualisierte Routingtabellen zur Routenermittlung. Im Gegensatz dazu verwendet es für ein Routing zwischen den Zonen den reaktiven Ansatz. Darauf aufbauend wurde neben vielen anderen Ansätzen das *Hierarchical State Routing* (HSR) [91] entwickelt. Dies ist ein von eher militärischen Überlegungen motiviertes Routingschema. Es unterteilt das Netzwerk in Cluster und setzt zu deren Verwaltung eine mehrstufige Hierarchie aus logischen Koordinatoren ein, deren Instanzen über das physikalische Netzwerk verteilt sind. Jeder Cluster des untersten Hierarchielevels instanziiert einen Koordinator, der wiederum Mitglied des darüber liegenden Hierarchielevels ist. Für HSR gibt es weder Vorgaben für die Partitionierung des Netzwerks noch für die genaue Platzierung von Koordinatoren. Stattdessen wird angedeutet, dass die Netzunterteilung sowohl auf geographischen als auch funktionalen Abhängigkeiten beruhen kann. Die Koordinatorinstanzen werden bei HSR sowohl für das Routing als auch für die eigentliche Paketweiterleitung verwendet. Folglich erreicht ein Paket sein Ziel stets entlang der physikalischen Wege, welche durch die aktuelle Hierarchiestruktur festgelegt werden. In früheren Providernetzwerken wurde ein ähnliches Routing in Form von PNNI angewandt. Nähere Details sind in Abschnitt 2.2.6.3 zu finden.

In heutigen Providernetzwerken und dem Kernnetzwerk des Internets wird durch OSPF beziehungsweise BGP ebenfalls eine zweistufige Hierarchie unterstützt. Die Aufteilung erfolgt dabei manuell durch den Netzwerkoperator. Beide verwenden ebenfalls eine Zielaggregation, sodass Routingtabellen heutiger Internetrouter stets aggregierte Routen zu großen Netzwerken enthalten.

2.3.4 Modulares Routing: Forwarding on Gates

Ab Oktobers 2008 finanzierte das Bundesministerium für Bildung und Forschung (BMBF) verschiedene über Deutschland verteilte Forschungsaktivitäten für ein zukünftiges Internet. Dazu zählte ab September 2009 ebenfalls das Projekt *G-Lab_FoG* (Projektnummer 16BK0935), welches die Erforschung des Ansatzes *Forwarding on Gates* (FoG) [9] [92] [93] zum Gegenstand hatte. FoG stellt eine vollwertige

Alternative zum heutigen Internet-Protokoll dar. Im Rahmen der Forschungsarbeit wurde die Software „FoG-Simulator/Emulator“ (*FoGSiEm*) entwickelt, die FoG vollständig implementiert. Im Gegensatz zu anderen Ansätzen für ein zukünftiges Internet, beispielsweise *Netlets* [94] oder *Service Oriented Node Architecture* (*SONATE*) [95] [96], war Routing in G-Lab_FoG ebenfalls ein Randthema. FoG ermöglicht durch seine Spezifikation den Einsatz verschiedener Routingansätze. Dabei sind jegliche Zwischenschritte zwischen quellbasiertem und verteiltem *Hop-by-Hop*-Routing möglich. Orthogonal dazu sind sowohl flache als auch hierarchische Managementstrukturen für FoG-Netzwerke anwendbar. Des Weiteren ist es möglich, für verschiedene Netzwerkabschnitte eigenständige Routingimplementierungen zu verwenden und diese für ein abschnittsübergreifendes Routing miteinander zu kombinieren. Aufgrund der beschriebenen Freiheitsgrade in FoG-Netzwerken wird in dieser Arbeit der Begriff des modularen Routings verwendet.
Im Vergleich zu anderen Ansätzen, wird FoG an dieser Stelle detaillierter erläutert, da es die Basis für die experimentellen Evaluierungen dieser Arbeit darstellt. Im Vordergrund der Erläuterungen stehen die durch FoG angewandte Funktionsverteilung im Netzwerk sowie eine Charakterisierung der Schnittstelle zwischen Paketweiterleitung und Routing in FoG-basierten Netzwerken.

2.3.4.1 Funktionsverteilung heutiger IP-Netzwerke

In heutigen IP-basierten Netzwerken erfolgt ein Routing von Anwendungsdaten typischerweise zweistufig. Die Anwendungsinstanz auf dem Quellknoten identifiziert das Ziel ihrer Daten zum einen mit Hilfe der Adresse des Zielknotens und zum anderen wird die Anwendungsinstanz auf Empfängerseite durch eine Portnummer festgelegt. Entsprechend Abschnitt 2.1.3 stellt dies eine Adressierung auf den Schichten 3 und 4 des OSI-Modells dar. Wie in Abschnitt 2.1.5.3 beschrieben, wird dafür bei IP jedem Knoten pro Netzwerkschnittstelle eine eigene Adresse zugeordnet, sodass er über sie als mögliches Ziel von Anwendungsdaten identifizierbar ist. Die Portnummern werden auf der Clientseite einer Kommunikation knotenlokal verwaltet, die Portnummer einer Serverseite als bekannt vorausgesetzt wird.

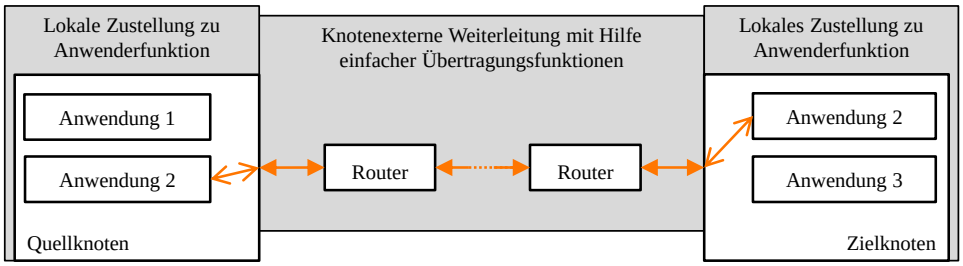

Abbildung 2.9: Übertragung von Anwendungsdaten in einem heutigen Netzwerk

Abbildung 2.9 stellt den resultierenden Übertragungsweg als orange markierte Pfeile für ein heutiges IP-basiertes Netzwerk dar. Er besteht aus drei Stufen für jedes Paket:

1.) Das Paket wird von der Anwendungsinstanz an die lokale Netzwerkschnittstelle übergeben.
2.) Mit Hilfe der Router im Netzwerk wird das Paket zum Zielknoten geführt.
3.) Auf dem Zielknoten wird das Paket der lokalen Anwendungsinstanz zugestellt.

Die Funktionen auf den Routern beschränken sich auf Routing und einfache Paketweiterleitung, sodass Pakete zwischen Knoten ausgetauscht werden können. Komplexere Funktionen befinden sich stattdessen entweder auf dem Quell- oder Zielknoten einer Datenübertragung.

2.3.4.2 Funktionsverteilung durch Gates und Weiterleitungsknoten

Im Gegensatz zur bekannten Unterteilung des Übertragungswegs in Knoten und Links, wendet FoG eine alternative Lösung an. Es werden sogenannte *Gates* eingeführt. Diese stellen einen unidirektionalen Übertragungsweg zwischen zwei *Weiterleitungsknoten* dar, die wiederum für die Übergabe eines Pakets von einem Gate an das folgende zuständig sind und somit auch als *Forwarding Node* (FN) bezeichnet

werden. Der Übertragungsweg eines Pakets besteht bei FoG somit aus einer Verkettung von Gates mit Hilfe von Weiterleitungsknoten. An den Enden der Kette befindet sich wiederum jeweils ein zusätzlicher Weiterleitungsknoten. Diese beiden stellen die Quell- und Zielanwendung dar.

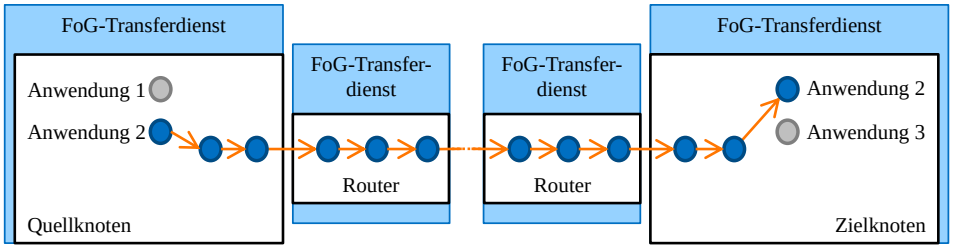

Abbildung 2.10: Übertragung von Anwendungsdaten in einem FoG-Netzwerk

Abbildung 2.10 stellt als Gegenstück zu Abbildung 2.9 den Übertragungsweg als orange markierte Pfeile für ein zukünftiges FoG-Netzwerk dar. Darin sind die an der Übertragung zwischen den Instanzen von Anwendung 2 aktiv beteiligten Weiterleitungsknoten in blau dargestellt. Mit grauer Farbe sind die unbeteiligten Weiterleitungsknoten der parallelen Instanzen von Anwendung 1 und 3 zu sehen. Verglichen mit Abbildung 2.9 besteht der Übertragungsweg auf einem Router bei FoG aus drei zusätzlichen Gates, wovon jedes die Aufgabe der Paketweiterleitung übernimmt und dadurch die Daten näher in Richtung ihres Ziels überträgt. Jeder physikalische Knoten besitzt dabei stets einen zentralen Weiterleitungsknoten, über den alle kontenlokalen Pfade miteinander verbunden sind.

Allgemein betrachtet besitzt ein Gate stets eine oder mehrere innere Funktionen. Gates stellen sogenannte funktionale Blöcke dar. Ein Block kann einerseits eine einfache Paketweiterleitung zum nächsten Weiterleitungsknoten realisieren, andererseits sind komplexere Prozesse möglich. Das kann beispielsweise eine Datenverschlüsselung oder eine Paketfilterung sein. Jedes Gate kann knotenlokal von anderen durch seine *Gatenummer* unterschieden werden. Diese muss für jeden Weiterleitungsknoten eindeutig sein, sodass durch eine Gatenummer der nächste zu passierende funktionale Block für ein Paket festgelegt ist.

2.3.4.3 Dienste

Die Signalisierungen und Aufgaben von FoG sind grundsätzlich in drei Dienste unterteilt. Dazu gehören:

- ***Transfer Service*:** Dies ist die Laufzeitumgebung für funktionale Blöcke. Dieser Dienst beinhaltet sowohl Gates als auch Weiterleitungsknoten. Abbildung 2.10 zeigt den Transferdienst für vier physikalische Knoten. Dieser Dienst ist ebenfalls für die automatische Erstellung von zusätzlichen Gates zuständig.
- ***Routing Service*:** Im Allgemeinen bestimmt dieser Dienst, welche vorhandenen Gates und Weiterleitungsknoten durch Pakete passiert werden müssen, um an ihr Ziel zu gelangen. FoG gibt für den Routingdienst keine Beschränkungen für die zu verwendende Struktur und internen Signalisierungen vor. Es sind sowohl flache als auch hierarchische Routingdienste möglich. Sollte der Routingdienst weitere Gates benötigen, löst er deren Erstellung im Transferdienst aus.
- ***Authentication Service*:** Die Aufgabe dieses Dienstes ist es, grundlegende Mechanismen zur Authentisierung und Autorisation sowie zum Accounting bereit zu stellen. Er wird an dieser Stelle zur Vollständigkeit erwähnt und steht nicht im Vordergrund dieser Arbeit.

Die Unterteilung in *Transfer Service* und *Routing Service* erlaubt eine strikte Separierung der einfachen Paketweiterleitung von den notwendigen Routingentscheidungen. Beide Dienste kommunizieren dabei über eine FoG-spezifische Signalisierung. Dies ermöglicht den Einsatz verschiedenster Routingdienste – und somit auch Routingalgorithmen – innerhalb eines FoG-Netzwerks.

2.3.4.4 Signalisierung von Anforderungen

Bei FoG kann die Routingentscheidung über sogenannte *Requirements* beeinflusst werden. Diese stellen Anforderungen der sendenden Anwendungsinstanz oder von Zwischenroutern dar, welche entlang des nachfolgenden Pfades zum Ziel einzuhalten sind. Eine Anwendung übergibt diese dem FoG-spezifischen Netzwerkstack zum Start einer neuen FoG-Verbindung. Wie im nachfolgenden Abschnitt beschrieben, werden diese Anforderungen innerhalb der verschickten Pakete an die Zwischenrouter des Netzwerks signalisiert.

2.3.4.5 Schnittstelle zwischen Paketweiterleitung und Routing

Ähnlich *Pathlet Routing* [97] unterstützt FoG ein fragmentiertes Routing. Dadurch sind in FoG-basierten Netzwerken alle Zwischenschritte zwischen quellbasiertem und *Hop-by-Hop*-Routing möglich, einschließlich der beiden genannten Ausprägungen am Rand des Spektrums. Im ersten Fall besteht das Routing aus genau einer Berechnung auf dem Quellknoten. Im letzten Fall wird eine Routingentscheidung auf jedem Zwischenknoten getroffen, bis das Paket am Zielknoten eingetroffen ist. Dies spiegelt sich im Aufbau von FoG-Paketen wieder. Die Metadaten innerhalb eines Paketkopfes beschreiben die Route für das Paket und werden vom Transferdienst während der Paketweiterleitung verwendet, um den jeweils nächsten Weiterleitungsknoten zu bestimmen. Dabei besteht die Route aus einer Liste von einem oder mehreren Segmenten der folgenden Typen:

- **Explizites Segment:** Das Segment beinhaltet eine Liste von Gatenummern, welche ein Fragment einer Route repräsentieren. Eine typische Darstellung ist beispielsweise „[4, 10]“, was in diesem Fall die Pakete entlang der Gates mit den Nummern 4 und 10 leiten würde.
- **Zielsegment:** In diesem Fall wird ein (Zwischen-)Ziel festgelegt, welches durch die Pakete passiert werden muss. Dabei wird durch FoG nicht vorgegeben, welches Format die Zielbeschreibung besitzen muss. Dadurch sind sowohl Namen als auch Adressen zur Zielbeschreibung möglich. Beispielsweise kann ein Knoten über eine global eindeutige Adresse der Form [71985f49-1ca1-11d3-9cc8-00c04f7971e0] als Zwischenrouter festgelegt werden. Zusätzlich kann ein Zielsegment auch die Requirements der sendenden Anwendung beinhalten. Alle Informationen eines Zielsegments werden bei einer Routinganfrage an den jeweiligen Routingdienst übermittelt.

Durch die Unterstützung von Zielsegmenten, kann eine erneute Routinganfrage explizit ausgelöst werden. Der jeweilige Routingdienst wird dazu verwendet, das Zielsegment in ein explizites Segment umzuwandeln, sodass die Paketweiterleitung fortgesetzt werden kann. Falls mehrere Zielsegmente in der Route enthalten sind, ergibt sich durch wiederholte Ausführung des beschriebenen Vorgangs das Gesamtrouting.

FoG unterstützt ebenfalls Qualitätsanforderungen während des Routings. Zu diesem Zweck beinhaltet das Konzept Mechanismen zur Festlegung einer geforderten Datenrate als auch der erlaubten Gesamtverzögerung für eine Übertragung. Diese Werte sind in Zielsegmenten innerhalb von FoG-Paketen wiederzufinden, sodass sie zwischen Transfer- und Routingdienst ausgetauscht werden können und für Routinganfragen auf Zwischenknoten zur Verfügung stehen.

2.3.4.6 Routing

Im Vergleich zu OSPF und BGP stellt FoG kein klassisches Routingprotokoll dar. FoG ist aus Sicht des Autors dieser Arbeit stattdessen als Rahmenwerk für den Einsatz von verschiedenen Implementierungen zu sehen. Für den Vergleich zwischen verschiedensten Routingvarianten sind insbesondere die Möglichkeiten interessant, Routingentscheidungen sowohl auf einem Knoten, auf ausgewählten Zwischenknoten als auch auf jedem Knoten entlang einer Route berechnen lassen zu können. FoG eignet sich daher als Basis für Vergleiche zwischen zukünftigen Routingprotokollen sowie bisher bekannten Implementierungen. Zum Zeitpunkt dieser Arbeit existierte bereits neben einer BGP-Integration auch ein

simulierter Routingdienst für FoG, welcher kein eigenes Signalisierungsprotokoll beinhaltet. Er unterscheidet sich jedoch grundsätzlich von BGP und ist somit relevant für die spätere Einordnung des in dieser Arbeit vorgestellten Konzeptes.

Eigenschaft	Simulierter Routingdienst
Einordnung der Signalisierungen im OSI-Modell	kA
Sicherung von Signalisierungen	kA
Clusterbildung	ja (manuelle Eingabe)
Hierarchiebildung	ja (manuelle Eingabe)
Adresszuweisung	kA
Zielaggregation	kA
Verteilung von Routingdaten	kA
Metrik	Hop-Distanz
Routingstrategie	*Shortest Path*
Beachtung von Qualitätsanf. beim Routing	nein
Routenberechnung	*Dijkstra*-Algorithmus
Routingzeitpunkt	reaktives Routing
Möglichkeit zur Vermeidung von Routingschleifen	ja

Tabelle 2.9: Eigenschaften des simulierten Routingdienstes für FoG

In Tabelle 2.9 wird der simulierte Routingdienst von FoG charakterisiert, dadurch wird ein direkter Vergleich zu Tabelle 2.8 aus Abschnitt 2.1.8.4 ermöglicht. Einige Eigenschaften können für den simulierten Routingdienst dabei nicht bestimmt werden, da die verfügbare Implementierung ohne Signalisierungen im Netzwerk auf Basis von lokalen Funktionsaufrufen arbeitet. Die Topologie wird dabei an eine für den jeweiligen Netzwerkabschnitt zentrale Routingdienstinstanz per direktem Funktionsaufruf übermittelt. Diese kann nachfolgend mit Hilfe des *Dijkstra*-Algorithmus die jeweils kürzeste Route zum gewünschten Ziel ermitteln.

2.4 Schlussfolgerungen

Kapitel 2 erläuterte die wichtigsten bereits existierenden Routingprotokolle für Intra-AS- und Inter-AS-Routing. Des Weiteren sind Erweiterungen für OSPF als auch BGP zur Beachtung von Qualitätsanforderungen bei Routingentscheidungen in heutigen Netzwerken vorgestellt wurden. Das Kapitel schließt mit ausgewählten Forschungsarbeiten, zu denen insbesondere HSR, *Selective Probes* als auch FoG zählen. Das Kapitel verdeutlicht, dass existierende Lösungen häufig auf Konfigurationsparametern beruhen, welche manuell durch den Netzwerkadministrator eingestellt werden müssen. Heutige Routinglösungen, wie OSPF oder BGP, beruhen auf der Annahme, dass sowohl die Gruppierung von Knoten als auch die dazugehörige Vergabe von Adressen vorgegeben sind. Diese Schritte werden typischerweise manuell durch den Netzwerkoperator durchgeführt.

3 Hierarchisches Routingmanagement

Ausgehend von den Beschreibungen in Kapitel 2 zu bekannten Konzepten der paketbasierten Datenübertragung und den dabei aufgedeckten Lücken für eine autonome Arbeitsweise beschreibt dieses Kapitel das neue Konzept des *Hierarchischen Routingmanagements* (HRM). Es unterteilt das zugrundeliegende Netzwerk und instanziiert auf den Knoten eine Managementinfrastruktur. Mit Hilfe dieser Instanzen werden Adressen und Routingdaten im Netzwerk verteilt. Dadurch ermöglicht HRM ein Routing von Paketen unter Beachtung der Anforderungen an die resultierende Übertragung von Anwendungsdaten. HRM verwendet für seine internen Signalisierungen ausschließlich autonom ablaufende Prozesse und ist dadurch in der Lage, ohne manuelle Eingaben ein Netzwerk zu verwalten und Anwendungsdaten an ihr jeweiliges Ziel zu leiten. Dieser Punkt unterscheidet HRM grundsätzlich von bekannten Lösungen.

Die ersten Überlegungen zu HRM begannen im Jahr 2009. Den größten Einfluss übte dabei der Ansatz *Hierarchical State Routing* aus Abschnitt 2.3.3 auf das Konzept aus. Die dabei entstandenen theoretischen Grundlagen wurden in [98] im Jahr 2010 verfeinert, sodass anschließend eine erste Implementierung und Evaluierung in [99] möglich war. Die darin verwendeten Algorithmen zum Clustern von Netzwerken und zur Ausbildung einer Managementhierarchie wurden anschließend vom Autor der vorliegenden Arbeit grundsätzlich überarbeitet. Das daraus entstandene Konzept ist Gegenstand dieses Kapitels, wobei ausgewählte Teile bereits in [100] öffentlich präsentiert worden sind.

Dieses Kapitel wird im nachfolgenden Abschnitt 3.1 mit einem Überblick über die zu erfüllenden Anforderungen an das neue Routingmanagement fortgesetzt. Im Anschluss wird in Abschnitt 3.2 die Architektur des neuen Routingmanagements als Antwort auf die zuvor aufgestellten Anforderungen vorgestellt. Die anschließenden Abschnitte 3.3 bis einschließlich 3.8 richten den Fokus auf die detaillierten Abläufe innerhalb der Architektur und erläutern diese jeweils anhand eines ausgewählten Beispielnetzwerks. Dieses besteht aus acht im Kreis angeordneten Knoten (Ringtopologie) und wird für alle Erklärungen als Referenzszenario[1] verwendet. Im Vordergrund der Erläuterungen stehen die verwendeten Prozesse sowie die verwendeten Signalisierungen, alle darin verwendeten Nachrichtentypen und die jeweils enthaltenen Daten sind zusätzlich in Anhang B tabellarisch zusammengefasst. Dabei wird ebenfalls das durch die Datenebene bereitgestellte QoS-Routing beschrieben. In Abschnitt 3.9 werden die Möglichkeiten zur Integration von HRM in heutige IP-basierte Netzwerke vorgestellt. Zum Ende von Kapitel 3 wird die beschriebene Architektur im Abschnitt 3.10 diskutiert, dabei wird sie mit den in Abschnitt 3.1 aufgeführten Anforderungen verglichen und erläutert, inwiefern diese von der Architektur beachtet werden. Das Kapitel 3 wird innerhalb von Abschnitt 3.10.9 mit einem detaillierten Vergleich der Konzeption mit bisherigen Ansätzen abgeschlossen.

[1] Das Szenario wurde explizit eher einfach gewählt, um eine möglichst übersichtliche Darstellung innerhalb dieser Arbeit zu gewährleisten. Die gewählte Topologie eignet sich trotz ihrer begrenzten Ausmaße dennoch zur Beschreibung aller relevanten Abläufe, dabei sind insbesondere die enthaltenen redundanten Routen für die Beschreibung der Datenebene und des darin integrierten Routingalgorithmus wichtig.

3.1 Anforderungen an die Architektur

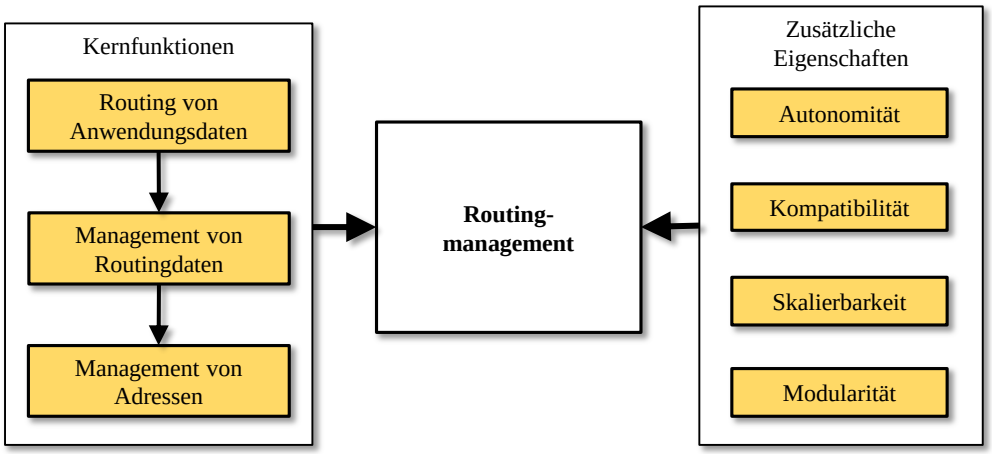

Abbildung 3.1: Anforderungen an das Routingmanagement

Bevor eine neue Architektur festgelegt werden kann, sind die Anforderungen zu definieren. Abbildung 3.1 gibt einen Überblick über die in dieser Arbeit gestellten Anforderungen an das Routingmanagement. Grundsätzlich werden dabei drei Kernfunktionen mit vier zusätzlichen Eigenschaften gefordert.

3.1.1 Kernfunktionen

Im Allgemeinen werden für das Routingmanagement drei wichtige Funktionskomponenten gefordert:

- **Routingalgorithmus:** Es muss einen Algorithmus geben, welcher für eintreffende Pakete die Routingentscheidung unter Beachtung der Qualitätsanforderungen der jeweiligen Anwendung trifft[2]. Entsprechend Abschnitt 2.2.3.4 muss er dabei die Datenrate und Verzögerung von bekannten Routen in seine Berechnungen einbeziehen.
- **Verteilung von Routingdaten:** Für korrekte Routingentscheidungen benötigt der Routingalgorithmus Daten über existierende Routen sowie zeitaktuelle Informationen über deren aktuell verfügbaren Datenraten sowie die zu erwartende Verzögerung bei Verwendung einer solchen Route. Diese Routingdaten müssen ausschließlich durch entsprechende Signalisierungen zwischen den Knoten des Netzwerks ausgetauscht werden und dienen bei anstehenden Routingentscheidungen dem Algorithmus als Eingabe.
- **Adresszuweisung:** Pakete können eine unbegrenzte Anzahl an Zwischenknoten passieren. Zur Beschreibung existierender Routen müssen Zwischenknoten eindeutig festlegt werden können. Des Weiteren muss das Ziel eines Pakets eindeutig identifizierbar sein. Dafür kann ein Knoten über eine seiner Netzwerkschnittstellen identifiziert werden. Im Kontext des Routingmanagements der vorliegenden Arbeit soll jeder Knoten ein mögliches Ziel für Pakete darstellen. Somit müssen allen Knoten eindeutige Adressen innerhalb des Netzwerks zugewiesen werden. Wie nachfolgend näher erläutert, steht dabei eine autonome Vergabe von Adressen im Vordergrund, sodass die für IP-basierte Netzwerke typischen Methoden nicht ausreichen.

Es ist eine chronologische Kausalität zwischen den drei geforderten Funktionskomponenten erkennbar. Der Routingalgorithmus benötigt das Ergebnis des Managements zur Verteilung von Routingdaten. Dieses wiederum benötigt bereits zugewiesene Adressen auf jedem Knoten des Netzwerks.

[2] Die eigentliche Übergabe des Pakets an den Nachbarknoten ist abhängig von der jeweiligen Implementierung und den dabei verwendeten Protokoll von Schicht 2, sodass dies nicht im Fokus der Architektur steht.

3.1.2 Zusätzliche Eigenschaften

Die Kernfunktionen des neuen Routingmanagements sollen unter folgenden zusätzlichen Rahmenbedingungen bereitgestellt werden:

- **Autonomität:** Im Vordergrund der Konzeption der notwendigen Prozesse steht insbesondere eine autonome Arbeitsweise (auch als *plug'n'play*- oder *out-of-the-box*-Eigenschaft zu bezeichnen):
  - Die Verteilung der für das Management notwendigen Kontrollinstanzen soll ohne manuelle Eingaben erfolgen, sodass die Instanzen automatisiert auf den Knoten des Netzwerks platziert werden. Bei Topologieänderungen im Netzwerk muss das Management in der Lage sein, sofort zu reagieren und entsprechend der Vorgaben des Platzierungsalgorithmus Instanzen im Netzwerk zu entfernen oder neue zu erstellen. Eine explizite Unterteilung durch manuellen Eingriff, wie dies beispielsweise bei der Festlegung von Netzwerkabschnitten in Form von *Areas* bei OSPF durchgeführt wird, muss vermieden werden.
  - Jegliche für Routingentscheidungen notwendige Daten dürfen ausschließlich durch autonom ablaufende Prozesse im Netzwerk verteilt werden. Dazu zählt sowohl das Management von Adressen als auch von den darauf aufbauenden Routingdaten.
- **Kompatibilität:** Als zweiter wichtiger Punkt wird die Kompatibilität des Signalisierungskonzeptes des Routingmanagements verstanden. Neue Architekturen sollten stets möglichst kompatibel zu bisherigen sein, für heutige Netzwerke bedeutet dies eine Kompatibilität mit IPv4 sowie IPv6. Dadurch wird der praktische Nutzen des neuen Routingmanagements untermauert.
- **Skalierbarkeit:** Die Skalierbarkeit der im Management eingesetzten Signalisierungen spielt eine weitere Schlüsselrolle für das Maß der erzielten Gesamtperformanz des Systems. Es ist dabei wichtig, dass das Gesamtsystem mindestens für die gewünschten Anwendungsfälle mit akzeptablen Kosten arbeitet. Abschnitt 2.2.1 ordnet die Unterstützung von Qualitätsanforderungen insbesondere den *Enterprise Routern* zu. Folglich ist es sinnvoll, eine Skalierbarkeit für große Firmennetzwerke zu fordern, welche gegenüber den Netzwerken auf Inter-AS-Ebene eine weitaus geringere Komplexität aufweisen. Dennoch spielt das Internet als Bindeglied zwischen verschiedenen Firmenstandorten eine Rolle, das System sollte auch für diese Szenarien einsetzbar sein.
- **Modularität:** Die Gesamtkomplexität des Routingmanagements soll über verschiedene eigenständige Protokolle verteilt sein, wobei protokollübergreifende Optimierungen vermieden werden sollen. Durch den resultierenden modularen Aufbau sollen einzelne Teile des Gesamtsystems einfach gegen alternative Lösungen ausgetauscht und mit ihnen verglichen werden können. Dies soll zukünftigen Erweiterungen oder Anpassungen des Systems dienen.

3.2 Architekturüberblick

In den folgenden Abschnitten wird die Architektur des neuen Routingmanagements beschrieben. Begonnen wird mit einem Überblick über das verwendete Design der Architektur, es werden die notwendigen Komponenten und ihre jeweiligen Prozesse erläutert. Anschließend wird ein Überblick über die verwendete Verteilung von notwendigen Managementinstanzen gegeben, durch die die einzelnen Prozesse ausgeführt werden.

3.2.1 Grundlegendes Design und notwendige Prozesse

Für ein Design stellt sich in der ersten Überlegung die Frage, welche grundlegende Modularisierung der Funktionen sinnvoll ist. Entsprechend Abschnitt 2.1.8 kann ein Routingprotokoll in zwei Kernfunktionen unterteilt werden. Dies sind zum einen die notwendigen Signalisierungen zur Verteilung von Routingdaten und zum anderen sind das die eigentlichen Routingentscheidungen. Für Erstere werden stets entsprechende Kontrollinstanzen benötigt, welche die Daten über existierende Routen unter den Knoten

des Netzwerks verteilen. Das daraus resultierende Wissen über existierende Routen des Netzwerks dient wiederum als Eingabe für die gewünschten Routingentscheidungen. Diese funktionale Zweiteilung spiegelt sich ebenfalls in den Typen von auftretenden Paketen im Netzwerk wider: es gibt Signalisierungspakete, welche zwischen den Kontrollinstanzen des Routingmanagements ausgetauscht werden, und Datenpakete von Anwendungsinstanzen. Ohne Beschränkung der Allgemeinheit kann daraus eine Unterteilung des Routingmanagements in *Kontroll-* und *Datenebene* abgeleitet werden. Die Kontrollebene ist für notwendige Signalisierungen zur Verteilung von Routingdaten und den dafür benötigten Adressen sowie jeglichen zusätzlich notwendigen Managementdaten verantwortlich, während die *Datenebene* den Routingalgorithmus als Kernfunktion enthält. Dieser trifft die notwendigen Routingentscheidungen auf Basis von bekannten Routingdaten. Im Kontext von QoS-Routing wird dabei für Anwendungsdaten eine Route gewählt, welche die gegebenen Qualitätsanforderungen der jeweiligen Anwendung erfüllt.

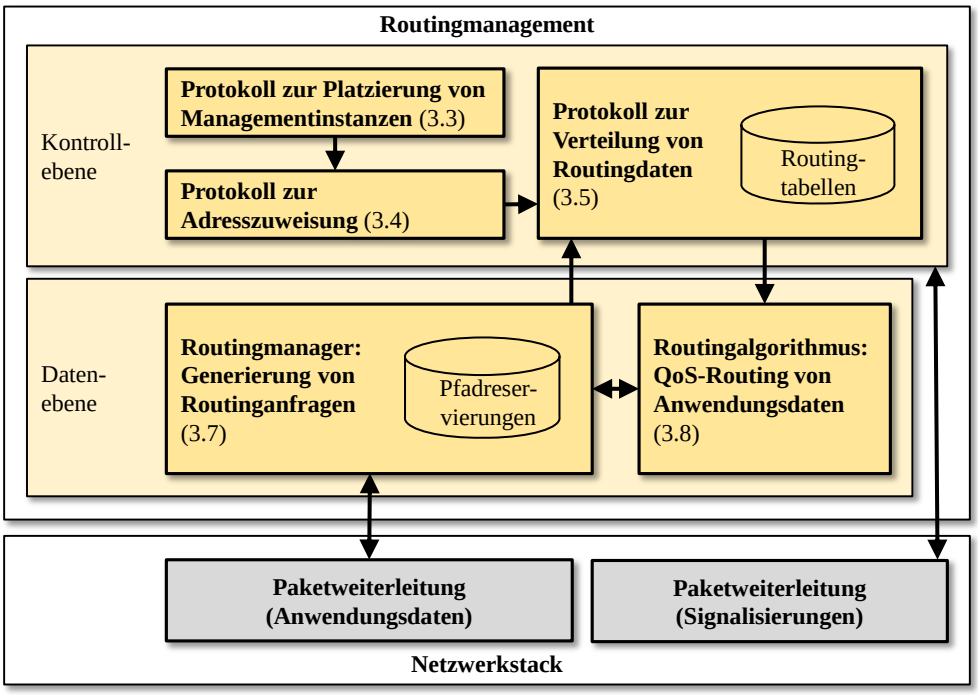

Abbildung 3.2: Architektur und wichtige Datenflüsse des Routingmanagements

Abbildung 3.2 stellt die Architektur des neuen Routingmanagements sowie ihre Interaktion mit dem zugrundeliegenden Netzwerkstack dar. In Klammern ist der zugehörige Abschnitt aufgeführt, welcher jeweils weitere Details enthält. Die Instanzen der globalen **Kontrollebene** wenden drei Protokolle an:

- **Protokoll zur Platzierung von Managementinstanzen:** Die notwendigen Entitäten der Kontrollebene müssen auf den Knoten des Netzwerks instanziiert werden. Dabei ist es nicht notwendig, dass jeder Knoten eine gleichartige Instanz beinhaltet. Folglich ist ein Signalisierungsprotokoll notwendig, worüber die Erstellung von Instanzen mit unterschiedlichem Typ synchronisiert wird. Die erstellten Entitäten besitzen dadurch für ausgewählte Bereiche des Netzwerks Kenntnis voneinander und können miteinander kommunizieren.
- **Protokoll zur Adresszuweisung:** Um Knoten als Zwischenstation einer Route oder als Ziel von Paketen identifizieren zu können, und vorhandene Routen zwischen ihnen beschreiben zu können, müssen den Knoten eindeutige Adressen zugeordnet werden. Zu diesem Zweck werden auf Basis von knotenübergreifenden Signalisierungen zwischen den Entitäten der Kontrollebene eindeutige Adressen im Netzwerk verteilt und den Knoten zugeordnet.

- **Protokoll zur Verteilung von Routingdaten:** Für dynamisches, verteiltes Routing müssen lokal auf jedem Router aktuelle Daten über die Topologie des Netzwerks vorliegen. Diese Routingdaten werden durch Signalisierungen zwischen den Instanzen der Kontrollebene verteilt, wodurch jeder Knoten seine lokale Routingtabelle ermittelt[3].

Auf Basis dieser drei Protokolle werden Adresszuweisungen durchgeführt und Routingdaten im Netzwerk verteilt, welche in Form von Routingtabellen auf jedem Knoten lokal gespeichert werden. Sie dienen als Eingabe des Routingalgorithmus der **Datenebene**, welche die notwendigen Routingentscheidungen für eintreffende Pakete bestimmt. Zu diesem Zweck führt jede lokal agierende Instanz der Datenebene zwei grundsätzliche Prozesse aus:

- **Routingmanager:** Dieser Manager existiert lokal auf jedem Knoten und prüft die Daten von eintreffenden Anwendungspaketen[4], daraus leitet er sowohl das jeweilige Ziel als auch die zugehörigen Qualitätsanforderungen ab. Insofern er keine für das jeweilige Paket zutreffende Pfadreservierung kennt, generiert er mit den zuvor bestimmten Daten eine Routinganfrage für den Routingalgorithmus. Dessen ermitteltes Ergebnis übergibt der Manager zusammen mit dem Paket an die weitere Paketverarbeitung durch das jeweils verwendete Protokoll von Schicht 3. Sollte dabei für den jeweiligen Datenstrom des Pakets noch keine Pfadreservierung existieren und eine neue erforderlich sein, werden durch den Manager die dafür notwendigen lokalen Ressourcen fest zugeordnet. In diesem Fall informiert er die knotenlokalen Instanzen der Kontrollebene, sodass diese aktualisierten Routingdaten an die anderen Managementinstanzen im Netzwerk signalisieren, wodurch alle Routingtabellen auf entfernten Knoten aktualisiert werden.
- **Routingalgorithmus:** Als Kernkomponente von HRM realisiert der Routingalgorithmus ein Routing unter Beachtung der Qualitätsanforderungen der jeweiligen Anwendung. Seine Eingaben erhält er zum einen vom Routingmanager und zum anderen aus der lokalen Routingtabelle, deren Inhalt durch die Kontrollebene stets aktuell gehalten wird.

Die eigentliche Paketübergabe an den Nachbarknoten verbleibt unverändert und wird durch das vorhandene Protokoll von Schicht 3 realisiert. Dies ist entweder IPv4/v6 oder ein ähnliches Protokoll zur paketbasierten Übertragung. Etwaige Datenkonvertierungen zwischen Schicht 3 und der Datenebene werden durch die jeweilige Implementierung des Routingmanagers durchgeführt.

3.2.2 Strukturierung der Kontrollebene

Die Protokolle der Kontrollebene verteilen unter den Knoten des Netzwerks die notwendigen Routingdaten für die Datenebene. Dabei ist die verwendete Struktur der Kontrollebene von entscheidender Bedeutung, um ein akzeptables Skalierungsverhalten der Signalisierungen für große Netzwerke zu ermöglichen. Eine schlechte Unterteilung kann die Gesamtperformanz des Systems für große Netzwerke signifikant negativ beeinflussen.

Ein weiterer Einflussfaktor für die Gesamtperformanz ist die Konvergenzzeit bis zur kompletten Verteilung von geänderten Routingdaten. Die Auswirkungen von Veränderungen der Netztopologie auf die Prozesse und der verursachte Signalisierungsaufwand sollten möglichst lokal begrenzt sein. Nach Analyse der Möglichkeiten zur Strukturierung einer Kontrollebene ergeben sich drei prinzipielle Varianten:

- **Gleichberechtigte Kontrollinstanzen:** Ein intuitiver Ansatz zur Verteilung von Routingdaten ist das Fluten des Netzwerks, wie es beispielsweise bei OSPF angewandt wird. Lokale Routingdaten werden dabei als Broadcast an alle Knoten gemeldet, wodurch jedoch insbesondere bei

[3] Auf Basis dieser Daten kann jeder Knoten ebenfalls seine lokale Weiterleitungstabelle bestimmen, dies ist jedoch nicht Bestandteil der Architektur.
[4] Diese können sowohl vom lokalen als auch von entfernten Knoten gesendet worden sein.

Topologieveränderungen ein hohes Datenaufkommen verursacht wird. Im Allgemeinen kommuniziert bei diesem Vorgehen jeder Knoten mit jedem anderen, sodass insgesamt $\frac{n*(n-1)}{2}$ Kommunikationsbeziehungen im Netzwerk entstehen, was einer quadratischen Komplexität von $O(n^2)$ entspricht.

- **Zentrale Kontrollinstanz:** Eine Verbesserung gegenüber der zuvor beschriebenen Signalisierung stellen die *Designated Router* von OSPF aus Abschnitt 2.1.8.2 dar. Sie werden durch den Netzwerkadministrator explizit festgelegt und übernehmen eine zentrale Koordinatorrolle während des Austauschs von Routingdaten, sodass alle Routingdaten ausschließlich an den zentralen *Designated Router* gemeldet und auch von ihm empfangen werden. Es ergeben sich $(n - 1)$ Kommunikationsbeziehungen, was zu einer linearen Gesamtkomplexität von $O(n)$ führt und somit erheblich geringer als die vorherige Kommunikationskomplexität von $O(n^2)$ ausfällt.
- **Baumstruktur aus Kontrollinstanzen:** Alternativ zur flachen Hierarchie mit einer zentralen Kontrollinstanz kann eine Baumstruktur angewandt werden, bestehend aus Kontrollinstanzen auf unterschiedlichen Hierarchielevels. Dabei wird jeder Kontrollinstanz exakt eine übergeordnete Instanz zugeordnet. Es entsteht eine mehrstufige Struktur, wobei sich an der Wurzel des Baumes eine zentrale Kontrollinstanz befindet. Daraus ergeben sich weitere Vorteile:
    - **Skalierbare Verteilung von Routingdaten:** Jede Kontrollinstanz in den Blättern des Baumes ist nur für einen kleinen Teilbereich des Netzwerks zuständig und kommuniziert ausschließlich mit ihrer jeweils übergeordneten Instanz. Die resultierende Kommunikationsinfrastruktur kann dazu verwendet werden, um von den Blättern ausgehend in Richtung der Wurzel des Baumes jeder Kontrollinstanz eine Übersicht über die jeweils untergeordneten Teilbereiche des Netzwerks mitzuteilen. Diese erhaltenen Topologiedaten können aggregiert und an die anderen untergeordneten Kontrollinstanzen signalisiert werden, wodurch Routingdaten im gesamten Netzwerk verteilt werden. Als Resultat steht jeder Kontrollinstanz (und somit auch jedem Knoten) zusätzlich eine aggregierte Sicht über entfernte Netzwerkabschnitte zur Verfügung. Durch die Anwendung einer solchen Aggregation kann der verursachte zusätzliche Datenaufwand klein gehalten werden.
    - **Verteilung von Berechnungen:** Aufgrund der Unterteilung des Netzwerks in Teilbereiche und der Unterteilung des Managements in verschiedene Hierarchielevels, erstreckt sich die Kontrollebene über die physikalischen Knoten als verteilt arbeitendes, logisches Netzwerk aus Kontrollinstanzen. Ableitungen neuer Routen durch Kombination bekannter Teilstücke, welche durch die Signalisierung von Routingdaten bekannt sind, erfolgen dabei parallel und unabhängig durch verschiedene Kontrollinstanzen. Dadurch wird die verursachte Last zur Speicherung der Netzwerktopologie und der notwendigen Routenberechnungen auf mehrere Knoten verteilt.
    - **Lokale Kommunikation:** Die Kontrollinstanzen in den Blättern des Baumes kommunizieren häufig nur mit übergeordneten Instanzen in ihrer lokalen Nachbarschaft, sodass dadurch häufig kurze Wege verwendet werden. Im Gegensatz dazu treten zwischen höheren Kontrollinstanzen typischerweise längere Wege auf, wobei ihre Menge mit zunehmendem Hierarchielevel stetig abnimmt. Somit ist bei geeigneter Strukturierung der Hierarchie zu erwarten, dass die durchschnittliche Weglänge für eine Aktualisierung von Routingdaten im Vergleich zum zentralen Ansatz geringer ausfällt.
    - **Begrenzte Wirkung von Ausfällen:** Fällt eine Kontrollinstanz in den Blättern des Baumes aus, beeinflusst dieser Ausfall nur lokale Bereiche der Kontrollebene. Anders verhält es sich für höhere Kontrollinstanzen. Hier gilt es, diese auf möglichst ausfallsicheren Knoten zu platzieren oder Ausfällen durch redundante Systeme vorzubeugen.

Aufgrund der charakteristischen Eigenschaften der verschiedenen Möglichkeiten zur Strukturierung der Kontrollebene verwendet das in dieser Arbeit vorgestellte Routingmanagement eine Baumstruktur mit einer mehrstufigen Hierarchie. Nachfolgend wird es als *Hierarchisches Routingmanagement* (HRM) bezeichnet und erfüllt die aus der Literatur bekannten Anforderungen an ein skalierbares Management von Adressierung sowie Routing:

> „As yet, there are no logical arguments (i.e., proofs) that a hierarchy is the only topology that can be used for large domains. On the other hand, we have no examples or proposals of any other topology actually being feasible on a large scale. This makes it difficult to make many general statements about addressing without assuming a hierarchical topology." (Kapitel 8 in [12])

> "The outlined principles for designing a scalable routing system are building routing hierarchy; introducing fault isolation; reducing routing processing burden where possible; defining manageable routing policies and using the assistance of available out-of-band routing process." (Kapitel 7 in [101])

Die Hierarchie der Kontrollebene wird bei HRM ausschließlich für die Signalisierung von Managementdaten verwendet. Sie bildet ein Overlay-Netzwerk oberhalb des physikalischen Netzwerks und verwaltet neben den Routingdaten auch die dafür notwendige Verteilung von Adressen sowie notwendige Statusaktualisierungen.

3.2.3 Strukturierung der Datenebene

Die Aufgabe der Datenebene ist es, Pakete von Anwendungen durch das Netzwerk zu leiten. Das Gesamtrouting vom Quell- zum Zielknoten besteht aus einzelnen Routingentscheidungen, dabei muss eine für jeden (Zwischen-)Knoten getroffen werden.

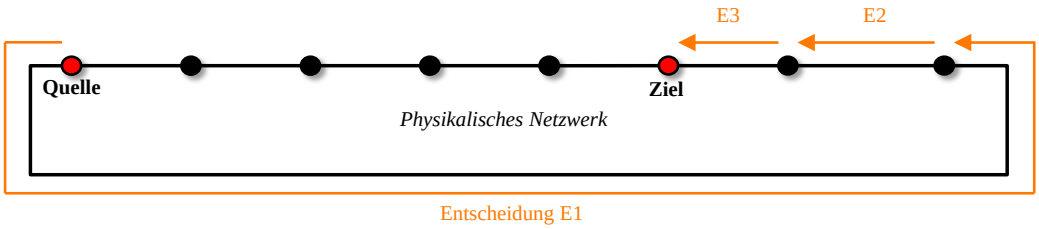

Abbildung 3.3: Einzelentscheidungen (orange) von Instanzen der Datenebene

Für das in Abbildung 3.3 dargestellte Beispielszenario werden für die orange dargestellte Route drei Routingentscheidungen benötigt, um ein Paket korrekt vom Quell- zum Zielknoten zu übertragen. Entsprechend Abschnitt 2.1.6 können die notwendigen Entscheidungen entweder durch eine zentrale Instanz (beispielsweise auf dem Quellknoten) oder durch verteilt platzierte Instanzen getroffen werden. Aufgrund seiner Vorteile wird bei HRM ein verteiltes Routing innerhalb der Datenebene favorisiert. Des Weiteren wird festgelegt, dass genau eine Instanz eines Routingmanagers der Datenebene auf jedem Knoten des Netzwerks existiert. Er leitet Routinganfragen ab und leitet diese an den knotenlokal ablaufenden Routingalgorithmus weiter. Es wird ein *Hop-by-Hop*-Routing angewandt, sodass jede Entscheidung immer nur den jeweils nächsten Knoten zum Ziel betrifft.

3.3 Protokoll zur Platzierung von Managementinstanzen

Als erster wichtiger Prozess von HRM wird bei einem Netzwerkstart die Instanziierung der Kontrollebene durchgeführt. Dadurch werden Kontrollinstanzen über das physikalische Netzwerk verteilt und auf unterschiedlichen Hierarchielevels angeordnet. Grundsätzlich wird zwischen dem Basislevel und den

darüber befindlichen höheren Hierarchielevels unterschieden. Jeder Kontrollinstanz liegt ein Cluster zugrunde, für den die erstellte Instanz sowohl Adressen als auch Routingdaten verteilt. Die Kontrollinstanzen nehmen folglich die Koordinatorrolle für den jeweiligen Cluster ein und werden aufgrund dessen nachfolgend als *Koordinatoren* der Kontrollebene bezeichnet. Innerhalb des Basislevels werden Cluster aus physikalischen Knoten gebildet, während in Clustern auf höheren Hierarchielevels die jeweiligen Koordinatoren des darunter liegenden Hierarchielevels zusammengenfasst werden.

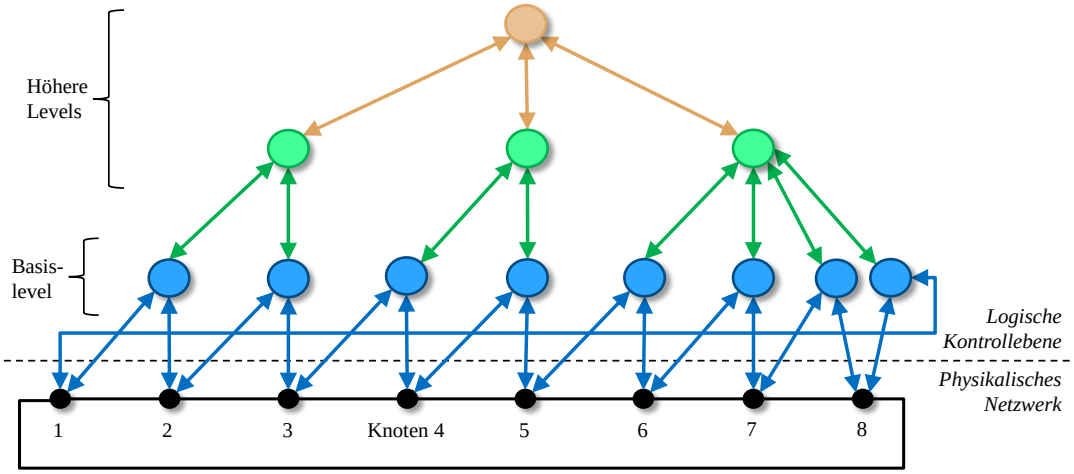

Abbildung 3.4: Platzierung der Kontrollinstanzen auf verschiedenen Hierarchielevels

Als Resultat des Erstellungsprozesses wird eine baumartige Struktur aus Instanzen der Kontrollebene ausgebildet, wie sie im oberen Teil von Abbildung 3.4 für eine Hierarchietiefe von 3 dargestellt ist. Zusätzlich zu dieser logischen Kontrollebene zeigt die Abbildung darunter das physikalische Netzwerk, bestehend aus der Ringtopologie mit acht Knoten, deren Nummerierung anhand der schwarzen Zahlen abzulesen ist. Innerhalb der logischen Kontrollebene ist die braun dargestellte Wurzel des Baumes in Form eines Koordinators auf Hierarchielevel 2 zu erkennen. Er bildet zusammen mit den darunter angeordneten Koordinatoren (grün dargestellt) des Hierarchielevels 1 die höheren Levels der Hierarchie. Die Blätter des Baumes werden wiederum durch die blau gekennzeichneten Kontrollinstanzen des Basislevels gebildet, welche jeweils einen Link verwalten und auf einem der beiden Knoten des jeweiligen Links platziert sind. Allgemein lässt sich anhand der horizontalen Anordnung der Koordinatoren in Abbildung 3.4 ebenfalls erkennen, dass auf einem physikalischen Knoten stets Koordinatoren von verschiedenen Hierarchielevels parallel existieren können.

In Abbildung 3.4 sind mit farbigen Pfeilen explizit die Kommunikationsbeziehungen zwischen den einzelnen Hierarchielevels dargestellt, sie bestehen jeweils zwischen einem Koordinator und den jeweiligen Mitgliedern seines Clusters. Innerhalb des Basislevels wird je physikalischem Knoten eine sogenannte Proxyinstanz verwendet, welche den jeweiligen Knoten innerhalb der Kommunikation der Kontrollebene repräsentiert. Zur Vereinfachung der Darstellung wurden diese Proxyinstanzen in dieser und auch den nachfolgenden Abbildungen weggelassen.

Die nachfolgenden Abschnitte erläutern, wie die Prozesse zur Strukturierung sowohl des Basislevels als auch der darüber befindlichen höheren Hierarchielevels allgemeingültig eine automatische Platzierung von Koordinatoren auf den physikalischen Knoten umsetzen.

3.3.1 Phase 0: Erkennung von Nachbarknoten

Die Erstellung der Kontrollebene startet mit der Bestimmung direkter Nachbarknoten. Auf dieser Basis kann nachfolgend das erste Level der gewünschten Hierarchie von Koordinatoren erstellt werden. Jeder Knoten benutzt dafür ein Hallo-Protokoll unter Verwendung der Broadcast-Adressen der jeweiligen lokalen Netzwerkschnittstelle, dazu wird das Protokoll von Schicht 2 verwendet. Die Signalisierungen des

Hallo-Protokolls werden nachfolgend als *AnnounceNeighborNode*[5]-Nachrichten bezeichnet, welche sowohl für die Anfrage als auch für deren Beantwortung verwendet werden. Ein Knoten sendet periodisch eine Anfrage an seine Nachbarn, dies wird von jedem Empfänger durch eine Antwortnachricht an den ursprünglichen Sender beantwortet. Sowohl die ursprüngliche Anfrage als auch deren Antworten enthalten eine eindeutige Nummer zur Identifikation des sendenden Knotens sowie eine eindeutige Anfragenummer zur Zuordnung eintreffender Antworten.

12345678-abcd-1234-abcd-123456789012

Abbildung 3.5: Beispiel eines *Universally Unique IDentifier* (UUID)

Die Identifikation eines Sendeknotens geschieht stets auf Basis einer sogenannten *Knoten-ID*. Sie kann beispielsweise aus einem *Universally Unique IDentifier* (UUID) [102] bestehen. Abbildung 3.5 zeigt ein Beispiel einer solchen UUID. Diese Form der Nummerierung wurde speziell für verteilte Systeme entwickelt und besteht aus 32 hexadezimalen Ziffern. Ein Knoten kann seine UUID lokal auf Basis lokaler MAC-Adressen und der Uhrzeit bestimmen, sodass eine Kollision zweier UUIDs nahezu ausgeschlossen werden kann. Eine zentrale Verwaltung von UUIDs und zusätzliche Signalisierungen werden dadurch nicht benötigt.

3.3.1.1 Topologieänderungen

Da sich die Topologie des Netzwerks stets ändern kann, ist die Menge lokaler Nachbarn ebenfalls dynamisch. Daten über direkte Nachbarn müssen folglich kontinuierlich aktualisiert werden. Zu diesem Zweck werden die *AnnounceNeighborNode*-Nachrichten nicht einmalig, sondern periodisch entsprechend eines definierten Intervalls $I_{probe\_neighborhood}$ versandt. Wird die *AnnounceNeighborNode* Nachricht durch einen Nachbarn beantwortet, kann dieser über seine im Antwortpaket enthaltene Knoten-ID identifiziert werden. Dadurch können ältere Nachbarn von neuen unterschieden werden. Bleibt aber eine Antwort auf eine *AnnounceNeighborNode*-Nachricht bis zum Erreichen einer lokal definierten Zeit $t_{invalid\_neighbor}$ aus, wird der Nachbarknoten als entfernt angenommen. Die Zeit $t_{invalid\_neighbor}$ wird dabei wie folgt berechnet:

$$t_{invalid\_neighbor} = t_{send\_time} + 2 * T_{delay\_E2E} + T_{receiver\_processing}$$

Formel 3.1: Berechnung der Zeit für die Löschung der Daten zu einem direkten Nachbarn

Die in Formel 3.1 verwendete Zeit t_{send_time} gibt die Zeit des Versendens der letzten *AnnounceNeighborNode* Nachricht an den jeweiligen Knoten an. Unter T_{delay_E2E} wird die erwartete maximale Ende-zu-Ende-Verzögerung zur Übertragung einer Nachricht zwischen zwei Knoten verstanden. Die auf dem Empfängerknoten benötigte Zeit zur Verarbeitung der Anfrage und Versenden der Antwortnachricht wird durch $T_{receiver\_processing}$ ausgedrückt.

Sollte unerwartet eine Netzwerkschnittstelle ausfallen, sind weitere *AnnounceNeighborNode* Nachrichten an ehemals darüber erreichbare Nachbarn überflüssig und diese Nachbarn können – bezogen auf diese Netzwerkschnittstelle – als entfernt angenommen werden. Über weitere alternative Links können diese Nachbarn jedoch weiterhin erreichbar sein. Der Status der Erreichbarkeit muss somit auf jedem Knoten pro Netzwerkschnittstelle gespeichert werden.

[5] Zur Vereinfachung der nachfolgenden Beschreibungen wird an dieser Stelle nicht näher auf den Unterschied zwischen Schicht 1 und 2 des OSI-Modells eingegangen. Ein über die Schicht 2 erreichbarer Nachknoten wird stets als physikalischer Endpunkt angesehen.

3.3.1.2 Ergebnis von Phase 0

Als Ergebnis der periodischen Signalisierung von Phase 0 sind jedem Knoten alle direkten Nachbarn bekannt. Switches, WLAN-Repeater und ähnliche Netzwerkgeräte für Schicht 2, welche typischerweise *store and forward* von Ethernet Frames realisieren, werden dabei nicht beachtet. Folglich gilt ein Hausnetzwerk, welches Knoten über verschiedene Switches von Schicht 2 des OSI-Modells miteinander verbindet, als eine Broadcast-Domäne. Alle Knoten sind in einem solchen Netzwerk direkte Nachbarn aus Sicht des Routings.

3.3.2 Phase 1: Strukturierung des Basislevels

Durch die Signalisierungen von Phase 0 liegen für Phase 1 die notwendigen Daten über alle Nachbarknoten der jeweiligen Broadcast-Domäne vor, sodass die Strukturierung von Hierarchielevel 0 der Kontrollebene beginnen kann. Nachfolgend wird dieses Level vereinfacht als L0 bezeichnet. Das Ziel von Phase 1 ist es, für jede Broadcast-Domäne einen L0-Koordinator automatisch auf einem der zugehörigen Knoten zu instanziieren.

3.3.2.1 Instanziierung von L0-Clustermanagern

Der Prozess von Phase 1 beginnt mit der Instanziierung von L0-Clustermanagern auf jedem Knoten. Für jede Broadcast-Domäne wird dabei auf jedem beteiligten Knoten eine Instanz erstellt, sodass auf jedem Knoten die Anzahl von existierenden L0-Clustermanagern der Anzahl von lokal vorhandenen Links entspricht. Jeder von ihnen kann aus den Signalisierungen von Phase 0 automatisch alle zugehörigen anderen Knoten der Domäne bestimmen, diese werden durch den jeweiligen L0-Clustermanager als L0-Cluster zusammengefasst. Die nachfolgende Abbildung zeigt die resultierende Struktur für das Beispielnetzwerk anhand des Links zwischen Knoten 4 und 5.

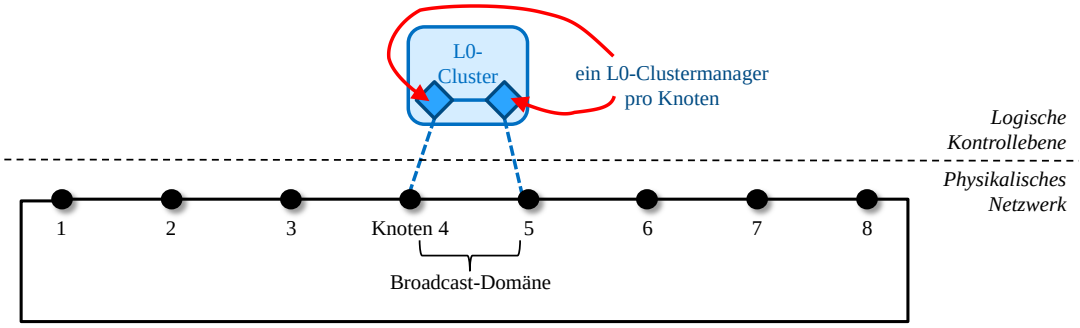

Abbildung 3.6: Beispiel eines L0-Clusters mit seinen beiden Clustermanagern als Mitglieder

In Abbildung 3.6 sind zwei L0-Clustermanager sowie der zugehörige L0-Cluster für eine ausgewählte Broadcast-Domäne (sie umfasst den Link zwischen Knoten 4 und 5) zu sehen. Auf jedem zugehörigen Knoten wurde ein eigenständiger L0-Clustermanager instanziiert. Dieser Prozess wird parallel dazu für jede Domäne durchgeführt, sodass für das Beispielnetzwerk in Abbildung 3.6 insgesamt je zwei L0-Clustermanager pro Knoten erstellt werden, sodass insgesamt 16 von ihnen im Netzwerk instanziiert werden[6].

Im nächsten Schritt werden aus den L0-Clustermanagern jene ausgewählt, welche eine lokale Koordinatorinstanz erstellen. Diese übernehmen für den jeweiligen L0-Cluster die zentrale Koordinatorrolle.

3.3.2.2 Instanziierung von L0-Koordinatoren

Zur Auswahl eines L0-Clustermanagers zur Instanziierung eines lokalen Koordinators muss ein Wahlalgorithmus eingesetzt werden. Dazu wird der bekannte Bully-Algorithmus [103] in angepasster Form

[6] Für eine Domäne mit mehr als 2 Knoten würde jeder der Knoten einen L0-Clustermanager für diese Domäne instanziieren, sodass bei 120 Knoten in einer Domäne insgesamt 120 L0-Clustermanager instanziiert werden.

zur Wahl von L0-Koordinatoren verwendet. Er arbeitet dezentral und bestimmt zuverlässig einen Koordinator aus einer Gruppe von Kandidaten. Der Algorithmus erfordert für seine Abläufe jedoch die Erfüllung von Annahmen. Dazu müssen jedem Wahlkandidat alle Alternativkandidaten sowie deren Identifikation bekannt sein. Dies ist aufgrund von Phase 0 für HRM gegeben: alle Nachbarknoten kennen einander und ihre Identifikation ist in Form der jeweiligen Knoten-ID signalisiert.

Da Koordinatoren innerhalb der Hierarchie untereinander kommunizieren, ist es sinnvoll, dass sie möglichst auf Knoten mit hoher Konnektivität platziert werden. Durch die resultierende zentrale Lage sind kurze Wege zu erwarten, wodurch die Belastung des Netzwerks durch Signalisierungen klein gehalten und somit die Gesamtperformanz des Gesamtsystems positiv beeinflusst wird. Um die Konnektivität in Wahlvorgängen zu beachten werden neben den konstanten Knoten-IDs zusätzlich die sogenannten Prioritäten als zusätzliches Kriterium verwendet. Dabei besitzt jeder Knoten seinen eigenen Wert, der ausschließlich von seiner Konnektivität abhängig ist und bei Topologieveränderungen automatisch angepasst wird. Ein solcher resultierender Wert wird nachfolgend als sogenannte *L0-Priorität* bezeichnet. Um die gewünschte Platzierung von Koordinatorinstanzen zu erreichen, werden Knoten mit hoher Konnektivität eine hohe L0-Priorität zugewiesen und bei Wahlvorgängen bevorzugt ausgewählt. Es ist sinnvoll, dass sich die L0-Priorität proportional der Konnektivität verhält, die L0-Priorität $p_{L0,k}$ eines Knotens k wird wie folgt bestimmt:

$$p_{L0,k} = c_k + w_k$$

Formel 3.2: Berechnung der L0-Priorität in Abhängigkeit von der Konnektivität

Die in Formel 3.2 verwendeten Werte ergeben sich aus:

- **c_k:** Aufgrund der Signalisierung von Phase 0 kennt jeder Knoten seine Konnektivität c_k. Sie gibt an, wie viele Knoten sich in direkter Nachbarschaft befinden. Durch diesen Parameter erhalten Knoten, welche sicher näher am Zentrum des Netzwerks befinden, eine höhere L0-Priorität.
- **w_k:** Diese Zahl gibt ein festgelegtes zusätzliches Gewicht für den Knoten k an, um Knoten eine höhere Priorität im Vergleich zu umliegenden Knoten zuzuordnen. Dies kann beispielsweise dazu verwendet werden, um Koordinatoren – unabhängig von der Topologie – auf Knoten mit überdurchschnittlichen Hardwareressourcen zu instanziieren. Der dafür notwendige Gewichtswert kann entweder manuell durch den Netzwerkadministrator oder automatisch durch ein externes System vergeben werden.

Die Prioritäten werden unter den Clustermanagern einer Domäne explizit signalisiert. Zu diesem Zweck wird der Nachrichtentyp *PriorityUpdate* eingeführt. Jeder L0-Clustermanager besitzt somit stets Kenntnis über die L0-Prioritäten aller anderen. Sollte sich die L0-Priorität eines Knotens aufgrund von Topologieänderungen ändern, sendet er den neuen Wert über seine L0-Clustermanager sofort mit Hilfe von *PriorityUpdate*-Nachrichten an alle anderen L0-Clustermanager von direkten Nachbarknoten.

Jeder Clustermanager entscheidet unabhängig von anderen Managern, wo die jeweilige Koordinatorinstanz platziert sein muss. Die dafür notwendigen Daten liefern die *PriorityUpdate*-Signalisierungen. Sollte ein Manager die höchste L0-Priorität besitzen, stellt er den Gewinner der Wahl dar, andernfalls ist ein anderer der Gewinner. Dabei ist es aufgrund der verwendeten Konnektivität c_k ebenfalls denkbar, dass zwei oder mehr L0-Clustermanager die gleiche L0-Priorität besitzen. In diesem Fall wird die in Phase 0 bereits übermittelte Knoten-ID als zusätzliches Vergleichskriterium verwendet. Da jede dieser IDs als eindeutig angenommen werden kann, führt ein Vergleich zwischen zwei Werten stets zu einem eindeutigen Ergebnis[7], der Clustermanager mit der höheren Knoten-ID ist in diesem Fall der Wahlsieger.

[7] Dabei wird eine äußerst geringe Kollisionswahrscheinlichkeit zwischen zwei berechneten Werten angenommen, was beispielsweise bei UUIDs gewährleistet ist.

Die nachfolgende Abbildung 3.7 zeigt den detaillierten Ablauf des Wahlvorganges unter Berücksichtigung von eintretenden Ereignissen.

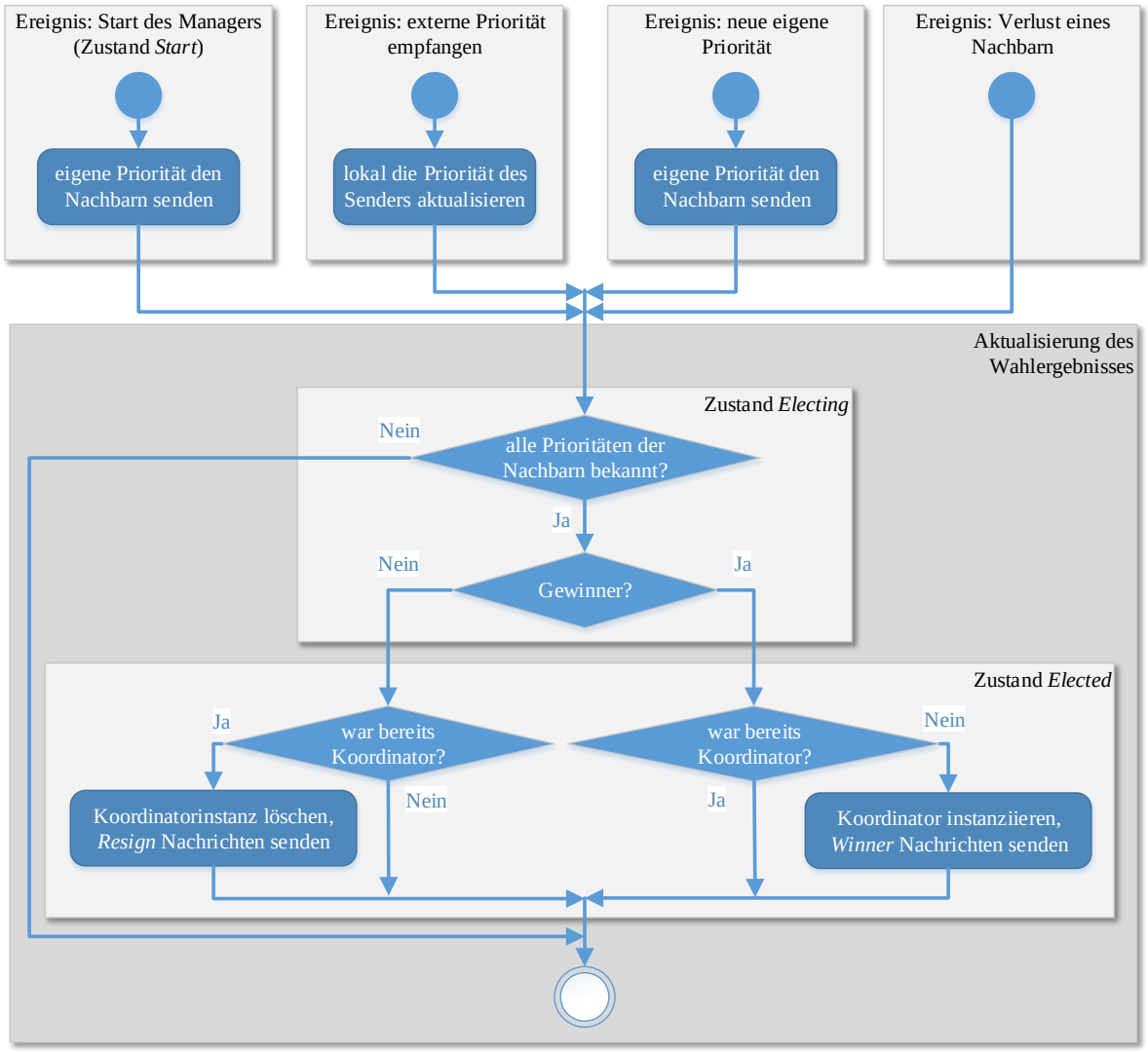

Abbildung 3.7: Reaktion eines L0 - Clustermanagers zur Aktualisierung des Wahlergebnisses

In Abbildung 3.7 ist zu erkennen, dass vier mögliche Ereignisse den Wahlvorgang[8] beeinflussen. Der lokale Start ist ein einmaliges Ereignis, während die restlichen drei Ereignisse zu jeder Zeit eintreten können. Der L0-Clustermanager reagiert jeweils mit einer Aktualisierung des lokalen Wahlergebnisses. Ähnlich dem Bully-Algorithmus reagiert ein L0-Clustermanager auf Hinzufügen oder Ausfall eines Teilnehmers mit einer erneuten Überprüfung des Wahlergebnisses. Dabei werden zur Erkennung von ausgefallenen Teilnehmern die periodischen Signalisierungen aus Phase 0 verwendet. Bei Ausbleiben von Antworten auf *AnnounceNeighborNode*-Nachrichten wird ein Nachbarknoten – und somit auch sein Clustermanager – als ausgefallen angenommen. Zusätzlich zu diesen Veränderungen an der grundsätzlichen Teilnehmermenge kann auch der Empfang einer neuen L0-Priorität die Ursache für ein neues Wahlergebnis sein. Der neue Wert kann entweder vom jeweiligen Knoten selbst oder einem seiner Nachbarknoten stammen. Dabei können externe Prioritäten durch Laufzeitverzögerungen während der Signalisierungen aus Phase 0 oder durch Veränderungen in der physikalischen Topologie verursacht werden. In beiden Fällen kann ein Clustermanager lokal den neuen Sieger ableiten. Wie in Abbildung

[8] Der dargestellte Ablauf repräsentiert die Reaktion auf jeweils ein eintretendes Ereignis, sodass der enthaltene Endzustand nicht dem Ende des eigentlichen Wahlvorganges gleichzusetzen ist.

3.7 zu sehen ist, teilt ein Wahlgewinner sein Ergebnis den anderen Clustermitgliedern über eine sogenannte *Winner*-Nachricht mit. Als Gegenstück dazu wird für HRM – im Gegensatz zum Bully-Algorithmus – der Nachrichtentyp *Resign* eingeführt. Er wird von einem Clustermanager an die anderen Teilnehmer der Domäne signalisiert, wenn er die Wahl verloren hat. Dieser Nachrichtentyp ist notwendig, um den Status der Instanziierung eines Koordinators anderen Teilnehmern mitzuteilen. Im Gegensatz zum Bully-Algorithmus sind die Prioritäten in HRM stets veränderlich. Dadurch kann ein Wahlgewinner zu jeder Zeit zum Wahlverlierer und umgekehrt werden. Wie in Abbildung 3.7 zu sehen ist, kann ein L0-Clustermanager einen der folgenden drei Zustände einnehmen:

- ***Start*:** Der Clustermanager wurde erstellt und dieser signalisiert seine Priorität.
- ***Electing***: Die Kommunikation mit Nachbarn wurde gestartet. Mindestens eine Priorität eines Nachbarn muss noch empfangen werden oder eine Prioritätsaktualisierung eines Nachbarn traf ein. Das Wahlergebnis liegt noch nicht vor.
- ***Elected***: Alle notwendigen Prioritäten liegen vor und das Wahlergebnis ist bestimmt. Der Clustermanager ist entweder der Gewinner oder der Verlierer der Wahl. Im ersten Fall existiert zusätzlich eine lokale Instanz eines L0-Koordinators.

Als Ergebnis des Wahlvorgangs ist für jeden L0-Cluster ein Koordinator instanziiert. Sollte eine Topologieänderung auftreten, wird diese durch veränderte L0-Prioritäten erfasst. Dies kann eine Neuplatzierung der jeweiligen Koordinatorinstanz zur Folge haben.

3.3.2.3 Vergleich mit dem Bully-Algorithmus

Der Bully-Algorithmus verwendet die Nachrichtentypen *Elect*, *Reply* und *Coordinator*[9]. Neue Wahlvorgänge werden über *Elect*-Nachrichten gestartet. Diese werden mit Hilfe von *Reply*-Nachrichten beantwortet. Nach Bestimmung des Wahlkandidaten mit der höchsten Identifikation sendet dieser Gewinner einmalig eine *Coordinator*-Nachricht an alle Wahlmitglieder, wodurch diese darüber informiert werden, dass er der Wahlsieger ist und ab jetzt als Koordinator zur Verfügung steht. Sollte ein Koordinator ausfallen, muss dies über einen externen Mechanismus erkannt werden. Der Bully-Algorithmus wird in diesem Fall erneut durch *Elect*-Nachrichten gestartet.

Ähnlich dem Bully-Algorithmus werden für HRM Knoten-IDs als eindeutige Identifikationen verwendet. Im Gegensatz zum Bully-Algorithmus werde diese jedoch mit den dynamischen L0-Prioritäten kombiniert, um den Gewinner eines Wahlvorganges zu bestimmen. Da die Prioritäten proportional zur Konnektivität der Knoten berechnet werden, werden Koordinatoren stets auf Knoten mit einer hohen Anzahl von Links zu direkten Nachbarn instanziiert. Ähnlich den *Coordinator*-Nachrichten des Bully-Algorithmus verwendet HRM die *Winner*-Nachrichten, um den Wahlsieger an die anderen Wahlteilnehmer zu signalisieren. Im Gegensatz zum Bully-Algorithmus wird bei HRM für Hierarchielevel 0 zusätzlich der Nachrichtentyp *PriorityUpdate* verwendet, mit dessen Hilfe wird die Dynamik von L0-Prioritäten realisiert. Die Werte aller Wahlmitglieder dürfen sich zu jeder Zeit verändern und werden automatisch über diesen Nachrichtentyp den Nachbarn mitgeteilt. Sollte ein Knoten oder Link ausfallen, bietet HRM zusätzlich einen Mechanismus zur Erkennung von Koordinatorausfällen. Dafür werden die periodischen Signalisierungen des Wahlgewinners verwendet, welche im nachfolgenden Abschnitt 3.3.6 detaillierter erläutert werden. Des Weiteren können durch die veränderlichen L0-Prioritäten bei HRM ebenfalls Situationen auftreten, in denen ein Wahlgewinner erneut zum Wahlverlierer wird. In diesem Fall verwendet HRM zusätzlich den Nachrichtentyp *Resign*, um dies an andere Wahlteilnehmer zu signalisieren.

[9] In der Literatur ist dieser Nachrichtentyp auch unter anderen Bezeichnungen zu finden (bspw. *Bully*).

3.3.2.4 Bekanntgabe von Koordinatoren

Nachdem ein L0-Koordinator instanziiert wurde, ist er prinzipiell für die Clustererstellung des übergeordneten Hierarchielevels 1 bereit. Ohne weitere Mechanismen würde keiner der umliegenden Knoten Kenntnis von der Existenz des L0-Koordinators besitzen. Folglich würde keiner von ihnen eine Kommunikation zum erstellten L0-Koordinator starten können. Um dies zu ermöglichen, müssen Daten über die Existenz eines jeden Koordinators im Netzwerk signalisiert werden. Dies geschieht in HRM auf Basis von sogenannten *AnnounceCoordinator*-Nachrichten. Ausgehend vom jeweiligen Koordinator werden diese Nachrichten konzentrisch in Form einer Welle im Netzwerk verbreitet.

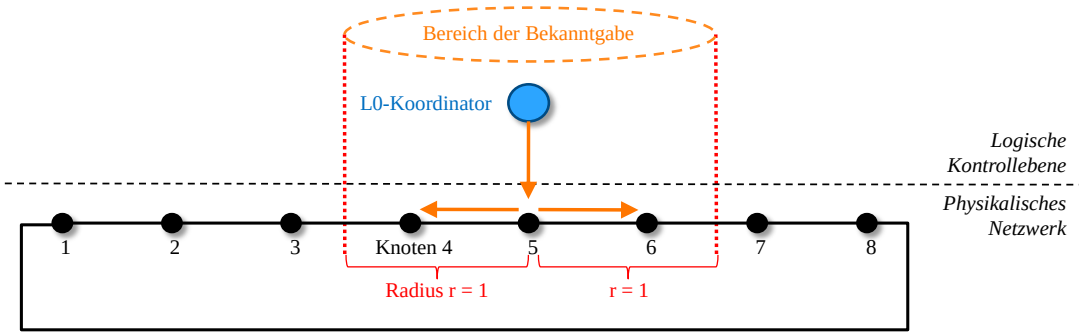

Abbildung 3.8: Bekanntgabe (orange) eines L0-Koordinators an Nachbarknoten in begrenzter (rot) Entfernung

Die Abbildung 3.8 zeigt einen Auszug aus der Kontrollebene. Es ist ausschließlich ein blau dargestellter L0-Koordinator zu sehen, die zugrundeliegenden L0-Cluster und ihre Clustermanager sind zur besseren Übersicht nicht abgebildet. Er wurde auf Basis des angepassten Bully-Algorithmus im Netzwerk instanziiert. Für seine Nutzung innerhalb des Hierarchielevels 1 ist eine Bekanntgabe in seiner lokalen Umgebung notwendig. Dies ist orange dargestellt. Die Nachricht wird ausschließlich innerhalb der örtlich begrenzten Nachbarschaft weitergeleitet. Dazu muss der Wert *r* global bekannt sein. Er gibt den maximal zulässigen Ausbreitungsbereich einer *AnnounceCoordinator*-Nachricht in Anzahl zulässiger Hops an. Im abgebildeten Beispiel ist der Radius mit 1 festgelegt, sodass sich die Nachricht über die Existenz des L0-Koordinators nur bis zu den direkten Nachbarknoten verbreitet.

Für eine korrekte Realisierung des Radius *r* für Level 0 muss innerhalb jeder *AnnounceCoordinator*-Nachricht die Anzahl bereits passierter Knoten in Form des *Hop-Zählers* mitgeführt werden. Trifft eine solche Nachricht an einem Knoten ein, muss dieser den Wert des *Hop-Zählers* um 1 erhöhen und die Nachricht an umliegende Knoten weiterleiten. Hierdurch erfahren alle Knoten einer begrenzten Nachbarschaft von der Existenz des L0-Koordinators.

3.3.2.5 Übertragung von Routingdaten und Schleifenerkennung

Ohne weitere Daten über eine erstellte Koordinatorinstanz sind entfernte Knoten nicht in der Lage, eine Kommunikation zu starten. Sie würden Kenntnis über die Existenz eines Koordinators besitzen, ihnen würde jedoch eine Route zum jeweiligen Knoten fehlen. Um dieses Problem zu lösen, sind zusätzliche Daten innerhalb von *AnnounceCoordinator*-Nachrichten notwendig. Während der Weiterleitung einer solchen Nachricht von Knoten zu Knoten wird die Rückwärtsroute aufgezeichnet. Sie wird als zusätzliches Element innerhalb der *AnnounceCoordinator*-Nachrichten gespeichert und beinhaltet die eindeutigen Knoten-IDs der Zwischenknoten, welche in umgekehrter Reihenfolge von einer Nachricht passiert werden müssen, um den Sendeknoten zu erreichen. Sollte es während einer *AnnounceCoordinator*-Signalisierung zu Topologieänderungen kommen, wird die veränderte Topologie nach Empfang der darauffolgenden *AnnounceCoordinator*-Nachricht für andere Knoten sichtbar.

Neben der Verwendung zum Routen von Signalisierungspaketen sind die übermittelten Rückwärtsrouten des Weiteren notwendig, um etwaige Schleifen in der Weiterleitung von *AnnounceCoordinator*-Nachrichten zu erkennen. Tritt eine Schleife in der Weiterleitung auf, findet ein Knoten seine eigene

Knoten-ID innerhalb der aufgezeichneten Rückwärtsroute wieder. In diesem Fall löscht er die Nachricht und verhindert dadurch eine redundante Übermittlung.

3.3.2.6 Annahmen des Wahlalgorithmus

Nachdem der Wahlalgorithmus erläutert wurde, gilt es die Frage zu beantworten, welche Annahmen vorausgesetzt werden müssen, um eine korrekte Arbeitsweise sicherzustellen. Zu diesen Annahmen zählen:

- **Korrektheit des Algorithmus:** Alle Knoten benutzen den gleichen Algorithmus.
- **Alle Kandidaten sind einander bekannt:** Durch die Signalisierung von Phase 0 sind jedem Clustermanager stets alle anderen Clustermanager bekannt.
- **Kooperatives Verhalten:** Jeder existierende L0-Clustermanager antwortet.
- **Ausschluss von Kommunikationsfehlern:** Keine der verschickten Nachrichten geht verloren oder wird auf Zwischenknoten verändert. Wie eine Vermeidung von Nachrichtenfehlern und -ausfällen realisiert wird, wird in Abschnitt 3.6.1.2 detaillierter behandelt. An dieser Stelle wird die Kommunikation als zuverlässig angenommen.
- **Konsistente Signalisierung:** Manipulierte Signalisierungen sind ausgeschlossen. Werden *AnnounceNeighborNode*-Nachrichten empfangen, existiert ihr Sender tatsächlich und ist der Sender der Nachricht.
- **Erkennung ausgefallener Kandidaten:** Durch Ausbleiben von *AnnounceNeighborNode*-Nachrichten aus Phase 0 wird ersichtlich, wenn ein Clustermanager auf einem entfernten Knoten ausgefallen ist. Er wird in dem Fall aus den Daten der anderen L0-Clustermanager der Broadcast-Domäne entfernt.
- **Totale Ordnung über die Kandidaten:** Allen L0- Clustermanagern wird eine L0-Priorität und eine eindeutige Knoten-ID zugeordnet. Während des Wahlvorganges werden beide Werte verwendet. Als Ergebnis wird stets eine eindeutige Ordnung unter den Mitgliedern eines L0-Clusters gefunden.
- **Deterministisches Verhalten des Algorithmus:** Ein Koordinator wird stets auf dem Knoten mit der für den jeweiligen L0-Cluster höchsten L0-Priorität bzw. Knoten-ID instanziiert.
- **Terminierung:** Der Algorithmus benötigt für die Bestimmung des Wahlergebnisses eine endliche Anzahl von Schritten.

Die aufgeführten Annahmen sind ähnlich denen des ursprünglichen Bully-Algorithmus (siehe Abschnitt 3.3.2.2).

3.3.2.7 Ergebnis von Phase 1

Durch Phase 1 wird das Basislevel der Hierarchie erstellt, dazu zählt ein L0-Cluster für jede Broadcast-Domäne. Für jeden dieser Cluster wird unter den zugehörigen Knoten durch den verwendeten Wahlalgorithmus einer selektiert, auf dem eine Koordinatorinstanz erstellt wird. Diese gibt anschließend ihre Existenz im umliegenden Netzwerk auf Basis von *AnnounceCoordinator*-Nachrichten bekannt. Dabei werden nur Knoten im Radius *r* benachrichtigt, sie erhalten durch diese Nachrichten ebenfalls eine Route zum sendenden Knoten.

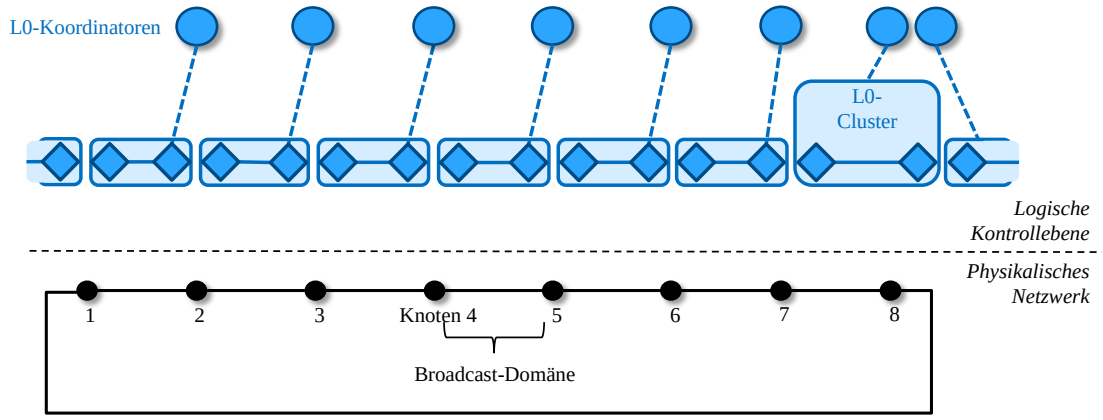

Abbildung 3.9: L0-Cluster und die Platzierung der L0-Koordinatoren in Abhängigkeit von den Knoten-IDs

Wendet man das beschriebene Konzept zur Strukturierung von Level 0 auf das Beispielszenario an, werden für die acht enthaltenen Knoten ebenfalls acht L0-Cluster mit je einem Koordinator instanziiert. In Abbildung 3.9 sind die resultierenden acht L0-Cluster als hellblaue Bereiche dargestellt, sie bestehen aus jeweils zwei Clustermanagern. Aufgrund der Ringtopologie ergibt sich für jeden Knoten die konstante L0-Priorität von 2, da jeder von ihnen exakt zwei Nachbarn besitzt. Die in Abbildung 3.9 schwarz dargestellten Zahlen stellen eine Nummerierung der Knoten dar und werden vereinfacht ebenfalls als Knoten-IDs verwendet. Aufgrund ihrer Einzigartigkeit stellen sie eine totale Ordnung unter allen Knoten sicher, sodass alle Koordinatoren deterministisch platziert werden. Die resultierenden L0-Koordiantoren sind in Abbildung 3.9 oberhalb der physikalischen Knoten als blaue kreisförmige Kreise abgebildet. Über die gestrichelten blauen Linien wird ihre Zugehörigkeit zu dem jeweiligen L0-Clustermanager angedeutet. Als Folge der höchsten Knoten-ID 8 werden auf dem letzten Knoten entsprechend zwei Koordinatoren platziert, da er für beide angrenzenden L0-Cluster die jeweils höchste L0-Priorität besitzt.

3.3.3 Paketbasierte Übertragung von Signalisierungen

Entsprechend den Anforderungen aus Abschnitt 3.1 an das Konzept von müssen die Signalisierungen der Kontrollebene sowohl autonom als auch kompatibel zu verwendeten Protokollen von Schicht 3 arbeiten. Dafür ist es notwendig, die typischen Funktionen zur paketbasierten Übertragung von Nachrichten der Kontrollebene näher zu betrachten. Entsprechend Kapitel 2 gehören dazu insbesondere die eingesetzte Adressierung sowie die Erstellung der für die Nachrichtenweiterleitung notwendigen Tabellen einzelner Knoten. Insbesondere für die nachfolgende Phase 2 muss es möglich sein, dass ein Clustermanager eines höheren Hierarchielevels mit einem untergeordneten Koordinator (und umgekehrt) kommunizieren kann, ohne dabei manuell vorgegebene Adressen oder Routen zu benötigen.

3.3.3.1 Adressierung Kontrollebene

Da auf einem Knoten stets mehrere Instanzen gleichen Typs auftreten können, muss es ebenfalls möglich sein, zwischen ihnen unterscheiden zu können. Dies betrifft sowohl Koordinator- als auch Clustermanagerinstanzen. Zur Lösung dieses Problems wird in HRM jeder Instanz eine eindeutige ID zugeordnet. Die Eindeutigkeit muss dabei knotenlokal sein, sodass die Verwaltung dieser IDs ebenfalls durch den jeweiligen Knoten durchgeführt wird.

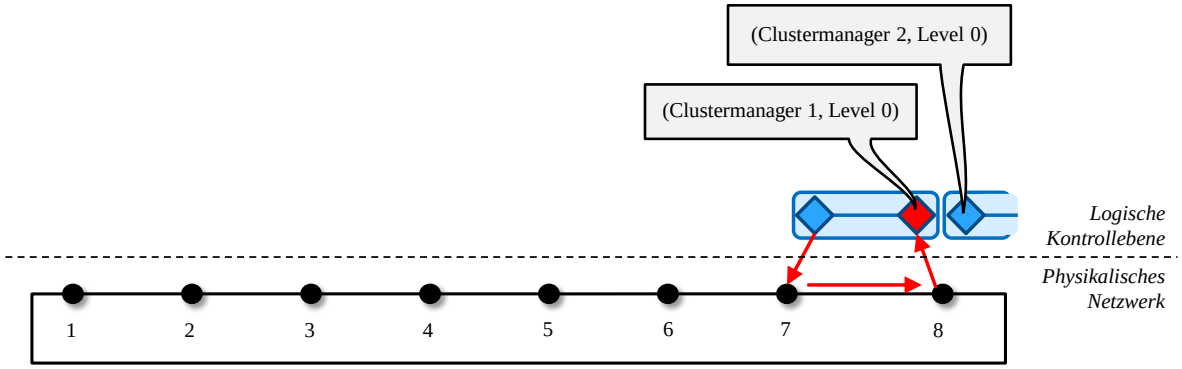

Abbildung 3.10: Multiple Cluster auf dem Zielknoten werden durch die Entität-ID adressiert

Angewandt auf das Beispielszenario kann eine Nachricht eindeutig an den gewünschten L0-Clustermanager auf Knoten 8 verschickt werden. Wie in Abbildung 3.10 erkennbar, sind beide Clustermanager auf Knoten 8 durch ein Tupel, bestehend aus Clustermanager-ID und Hierarchielevel, explizit adressierbar. Ein solches 2-Tupel wird im Folgenden als Entität-ID bezeichnet. Diese IDs stellen eine eigenständige Adressierungsebene dar und werden mit Hilfe der *AnnounceCoordinator*-Nachrichten von Phase 1 anderen Knoten mitgeteilt. Die Entität-IDs entfernter Instanzen sind dadurch stets bekannt. Dies ist insbesondere für die Clustererstellung auf höheren Hierarchielevels notwendig. Die Unterscheidung zwischen Clustermanagern und Koordinatoren ergibt sich dabei implizit aus dem jeweiligen Signalisierungskontext.

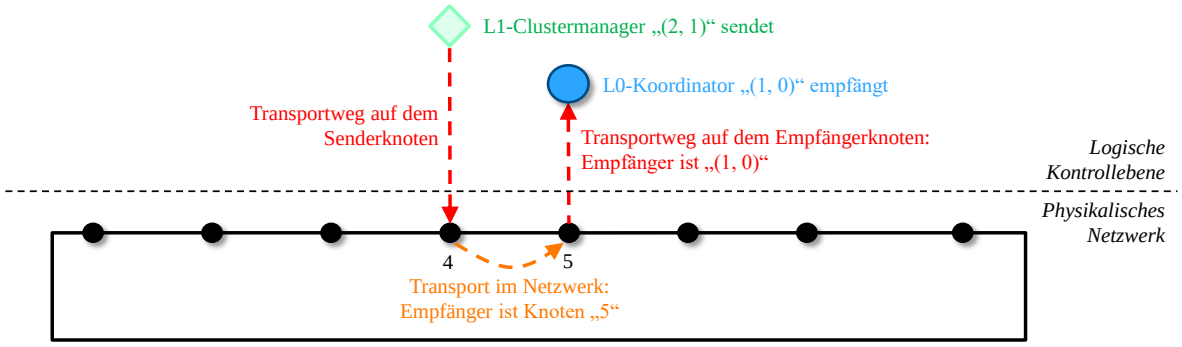

Abbildung 3.11: Signalisierungspakete werden über Knoten- und Entität-ID dem richtigen Empfänger zugestellt

Die Abbildung 3.11 zeigt die Verwendung von Entität-IDs am Beispiel der Kommunikation zwischen einem L1-Clustermanager auf Hierarchielevel 1 (als Beispiel eines Clustermanagers eines höheren Hierarchielevels) und einem untergeordnetem L0-Koordinator. Es ist zu sehen, dass der Sender der Signalisierung über seine Entität-ID *(2, 1)* zusammen mit der Knoten-ID *4* beschrieben wird. Der Empfänger wird mit Hilfe der Entität-ID *(1, 0)* und der Knoten-ID *5* adressiert. Die Entität-ID des Empfängers hat der Sender dabei aus vorhergehenden *AnnounceCoordinator*-Nachrichten entnommen. Auf Empfängerseite steht aufgrund des Typs der eintreffenden Nachrichten fest, dass die Signalisierung an eine lokale Koordinatorinstanz gerichtet ist. Die Nachricht wird der korrekten Entität auf Knoten 5 zugestellt[10].

3.3.3.2 Routingtabelle der Kontrollebene

Nachdem die Adressierung der Kontrollebene beleuchtet wurde, ist es wichtig zu klären, wie ein Signalisierungspaket einen Empfängerknoten erreichen kann, welcher mehr als einen Zwischenknoten entfernt liegt. Zu diesem Zweck werden die in einer *AnnounceCoordinator*-Nachricht aufgezeichnete Rückwärtsroute und der *Hop-Zähler* verwendet. Beide werden durch Phase 1 festgelegt und ermöglichen es jedem Knoten, kontinuierlich die jeweils kürzeste Route zum Sender lokal zu speichern. Die

[10] Interpretiert man Entitäten als Anwendungsinstanzen, entspricht eine Entität-ID der Portnummer auf dem jeweiligen Knoten.

daraus resultierende Routingtabelle wird ausschließlich für die Kommunikation innerhalb der Kontrollebene verwendet.

Ziel (Knoten-ID)	Nächster Knoten (Knoten-ID)	Hop-Distanz
[UUID 7]	[UUID 5]	3
[UUID 5]	[UUID 5]	1
[UUID 1]	[UUID 3]	3
[UUID 2]	[UUID 3]	1

Tabelle 3.1: Routingtabelle der Kontrollebene für Knoten 4 des Beispielszenarios

Tabelle 3.1 zeigt die resultierende Routingtabelle der Kontrollebene für Knoten 4 des Beispielnetzwerks. Der in der Tabelle orange dargestellte Eintrag beschreibt die in Abbildung 3.11 verwendete Route von Knoten 4 zu Knoten 5. Des Weiteren ist zu erkennen, dass pro Ziel jeweils nur der nächste Knoten einer Route gespeichert wird. Dies entspricht stets der kürzesten bekannten Route. Pakete der Kontrollebene werden folglich auf Basis von *Shortest Path Routing* durch das Netzwerk geleitet[11].

Sollten Linkausfälle auftreten, werden in *AnnounceCoordinator*-Nachrichten automatisch veränderte Routen zum Sendeknoten übermittelt. Die vorhergehende Route wird nicht mehr signalisiert und somit von den anderen Knoten nach Ablauf einer definierten Zeit automatisch als veraltet angenommen, sodass die Routingtabelle der Kontrollebene automatisch nach Empfang neuer *AnnounceCoordinator*-Nachrichten aktualisiert wird. Nähere Details dazu werden in Abschnitt 3.3.8 erläutert.

3.3.3.3 Weiterleitungstabelle der Kontrollebene

Wie in Abschnitt 2.1.7.2 beschrieben, ermittelt jeder Knoten aus der Routingtabelle seine lokale Weiterleitungstabelle. Mit deren Hilfe kann er für eintreffende Pakete die jeweilige Adresse für das Protokoll von Schicht 2 ermitteln und die Paketweiterleitung durchführen. Bei HRM werden dazu Knoten-IDs auf Adressen von Schicht 2 abgebildet, welche bei Ethernet beispielweise aus MAC-Adressen bestehen. Zu diesem Zweck prüft jeder Knoten die Daten von empfangenen *AnnounceNeighborNode*-Nachrichten und speichert die daraus gewonnenen Zuordnungen in seiner lokalen Weiterleitungstabelle ab.

3.3.4 Phase 2: Strukturierung von höheren Levels

Als Ergebnis von Phase 0 besitzt jeder Knoten Kenntnis über seine lokale Nachbarschaft. Aufgrund von Phase 1 wurde zudem für jede vorhandene Broadcast-Domäne eine L0-Koordinatorinstanz erstellt, welche später zur Verteilung von Routingdaten innerhalb ihrer jeweiligen Domäne genutzt wird. Für die domainübergreifende Verteilung werden jedoch weitere Koordinatorinstanzen benötigt. Phase 2 greift diesen Punkt auf und erstellt die höheren Levels der Kontrollebene, wobei die resultierende Struktur folgende Eigenschaften besitzen soll:

- **Anwendung einer Hierarchie:** Die Kontrollebene besteht aus einer Hierarchie, welche im obersten zulässigen Hierarchielevel durch einen *TOP-Koordinator* verwaltet wird, dieser Koordinator muss einzigartig für das gesamte Netzwerk sein.
- **Topologische Abhängigkeiten:** Ein Clustermanager auf Level *n* (und somit auch ein etwaiger Koordinator auf diesem Level) wird nur auf Knoten instanziiert, welche mindestens eine Koordinatorinstanz auf Level *(n - 1)* besitzen. Dies reduziert die Anzahl von instanziierten Clustermanagern und somit auch die Menge der parallelen Wahlvorgänge im Netzwerk sowie der dafür notwendigen Signalisierungen.

[11] Da *Shortest Path Routing* die Linkauslastungen nicht beachtet, sollten durch die Implementierung der Paketweiterleitung die Signalisierungspakete der Kontrollebene gegenüber Anwendungsdaten bevorzugt behandelt werden.

Die in diesem Abschnitt erläuterten Mechanismen können zur Bildung einer unbegrenzten Tiefe – und somit zu unbegrenzt vielen Levels der Hierarchie – genutzt werden. Zur Vereinfachung werden diese höheren Levels mit der Bezeichnung *Level 1+* oder *L1+* beschrieben. Stellvertretend für alle höheren Hierarchielevel wird im Folgenden der Algorithmus zur Strukturierung für Level 1 erläutert.

3.3.4.1 Skalierungsprobleme des Bully-Algorithmus

Im Allgemeinen könnte der Bully-Algorithmus verwendet werden, um einen Koordinator unter gleichberechtigten Kandidaten auszuwählen. Dieser Ansatz wird in Phase 1 eingesetzt, da davon auszugehen ist, dass sehr große Broadcast-Domänen eher untypisch für heutige Netzwerke sind. Für Phase 2 ist ein Einsatz des Bully-Algorithmus oder eines ähnlichen Ansatzes zu vermeiden. Die Gründe dafür werden im Folgenden erläutert.

Die Annahmen des Bully-Algorithmus sind in Abschnitt 3.3.2.2 erläutert. Er beruht darauf, dass sich alle Teilnehmer der Wahl kennen und miteinander kommunizieren. Bezogen auf ein Netzwerk aus *n* Knoten bedeutet das, dass jeder Knoten mit jedem anderen eine Signalisierungsverbindung aufrechterhalten muss. Daraus ergeben sich $\frac{n*(n-1)}{2}$ Verbindungen. Dies entspricht der allgemeinen Verbindungskomplexität $O(n^2)$. Bezogen auf die Strukturierung von Level 1 muss folglich jeder L1-Clustermanager auf Knoten mit L0-Koordinatoren die Routen zu allen anderen L1-Clustermanagern kennen. Auf Basis der in Phase 1 bereits verwendeten Kombination aus L0-Priorität und Knoten-IDs ist ebenfalls eine totale Ordnung über alle L1-Clustermanager durch entsprechende Signalisierungen ermittelbar. Dadurch kann stets ein L1-Koordinator bestimmt werden. Die resultierende Struktur von Level 1 ist in Abbildung 3.12 dargestellt.

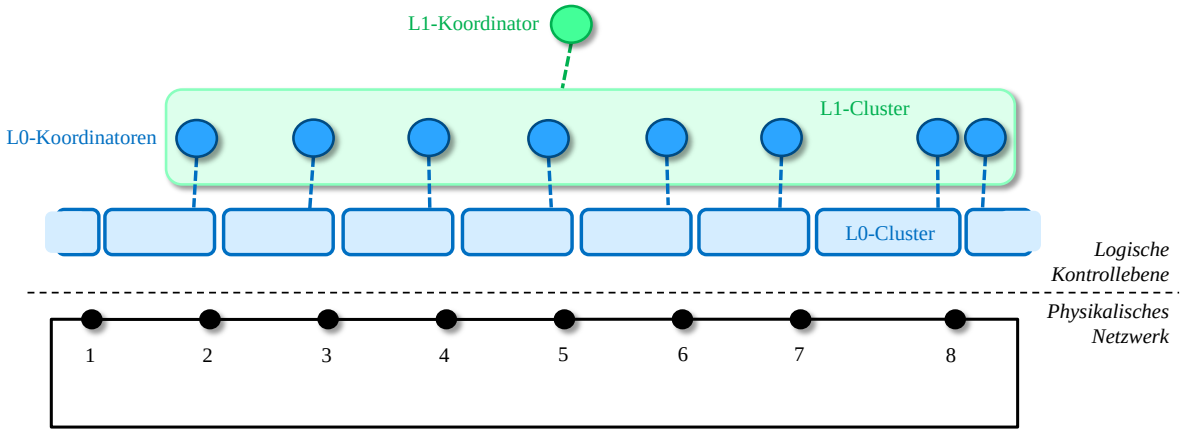

Abbildung 3.12: L1-Clusterbildung bei Verwendung des Bully-Algorithmus

In Abbildung 3.12 ist zu erkennen, dass die Anwendung des Bully-Algorithmus für sehr große Netze eine schlechte Skalierbarkeit zur Folge hat, der L1-Cluster umfasst stets alle untergeordneten L0-Koordinatoren und jeder Knoten muss jeden anderen Knoten sowie die Route zu ihm kennen. Des Weiteren muss eine Verbindung zwischen allen Knoten aufgebaut und für den L1-Koordinator aufrechterhalten werden. Um die Skalierbarkeit zu verbessern, ist eine manuelle Unterteilung von Level 1 denkbar. Dies widerspricht jedoch der Anforderung aus Abschnitt 3.1.2 nach autonomer Arbeitsweise der Kontrollebene. Der Bully-Algorithmus wird aufgrund seiner Annahmen, Vorbedingungen und schlechter Skalierbarkeit für große Netzwerke in HRM nicht für höhere Hierarchielevels eingesetzt.

3.3.4.2 Charakterisierung des Strukturierungsalgorithmus

Zur Realisierung einer automatischen Strukturierung höherer Hierarchielevels und einer darauf aufbauenden skalierbaren Verteilung von Routingdaten wird ein dezentraler Algorithmus eingeführt, welcher die folgenden Eigenschaften besitzt:

- **Clusterbildung:** Es werden topologisch begrenzte Cluster ausgebildet. Dabei wird ein global bekannter Wert *r* als maximaler Clusterradius verwendet. Durch diesen Wert wird die zulässige Höchstzahl an logischen Hops (bezogen auf das jeweilige Hierarchielevel) zwischen einem Clustermanager und den Mitgliedern seines Clusters festgelegt. Die Ausnahme für diese Begrenzung bildet dabei jedoch der Clustermanager auf dem höchsten Hierarchielevel, welcher die *TOP-Koordinatorinstanz* erstellt. Er muss mit allen untergeordneten Koordinatorinstanzen kommunizieren.
- **Autonome Arbeitsweise:** Die Unterteilung des Netzwerks geschieht ohne manuelle Eingaben. Bei Topologieänderungen wird die Struktur automatisch angepasst.
- **Dezentrale Arbeitsweise:** Der Algorithmus arbeitet ohne zentrale Koordinierungsinstanz im Netzwerk. Notwendige Daten werden pro Knoten jeweils unabhängig von den Aktivitäten anderer Knoten ermittelt.
- **Topologische Abhängigkeiten von Prioritäten:** Für jeden Knoten werden neben den L0-Prioritäten aus Phase 1 zusätzliche Prioritäten verwendet, welche für jedes Hierarchielevel verschieden sind. Eine Priorität für Level *n* ist dabei abhängig von der Topologie der Koordinatoren auf Hierarchielevel *(n - 1)*, sodass Koordinatorinstanzen auf Level *n* eher auf Knoten platziert werden, welche in ihrer unmittelbaren Nachbarschaft viele untergeordnete Instanzen besitzen.
- **Eindeutigkeit der Hierarchie:** Es gibt für jeden Koordinator auf Level *n* stets genau einen übergeordneten Koordinator auf Level *(n + 1)*, eine Zuordnung zu mehreren Koordinatoren des darüber liegenden Levels ist unzulässig. Sollten mehrere Koordinatoren auf Level *(n + 1)* existieren, wird der mit der höchsten Priorität zugeordnet.

Die Erstellung der Cluster auf höheren Hierarchielevels erfolgt *agglomerativ* [104], indem an den Blättern der Hierarchie mit sehr kleinen Clustern begonnen wird und die Struktur in Richtung der Spitze erstellt wird[12]. Dazu wird jeder Cluster zu Anfang mit einem Mitglied initialisiert und durch Hinzufügen weiterer Mitglieder nimmt er an Größe zu. Angewandt auf HRM bedeutet dies, dass jeder Knoten einen Cluster des jeweiligen Hierarchielevels bilden, der durch einen Clustermanager verwaltet wird. Erst durch Signalisierungen zwischen den Knoten bilden sich schrittweise größere Cluster aus, sie beinhalten untergeordnete Koordinatoren als Mitglieder. Dieser Algorithmus zur Strukturierung höherer Hierarchielevels wird im Folgenden schrittweise beschrieben.

3.3.4.3 Instanziierung von L1-Clustermanagern

Auf jedem Knoten, welcher eine Instanz eines Level *n* Koordinators besitzt, wird ebenfalls eine Instanz eines Level *(n + 1)* Clustermanagers erstellt. Sollte im Gegensatz dazu ein Clustermanager lokal keinen untergeordneten Koordinator mehr besitzen, wird er automatisch wieder entfernt.

Jeder dieser Clustermanager stellt einen potentiellen Kandidaten zur späteren Instanziierung eines Koordinators und damit auch zur Verwaltung von umgrenzenden untergeordneten Koordinatoren dar. Dies ist äquivalent zu einer Clustereinteilung untergeordneter Koordinatoren. Jeder der erstellten Clustermanager startet eine Kommunikation zu jedem ihm bekannten Koordinator des darunterliegenden Hierarchielevels durch Versand einer *RequestClusterMembership*-Nachricht. Als Eingabe benötigt dieser Schritt jeweils die Routingdaten aus Abschnitt 3.3.3 zu dem Knoten, auf welchem die Instanz des bekannten Level *n* Koordinators platziert ist.

[12] Die Erstellung der Hierarchie wird im Folgenden näher erläutert.

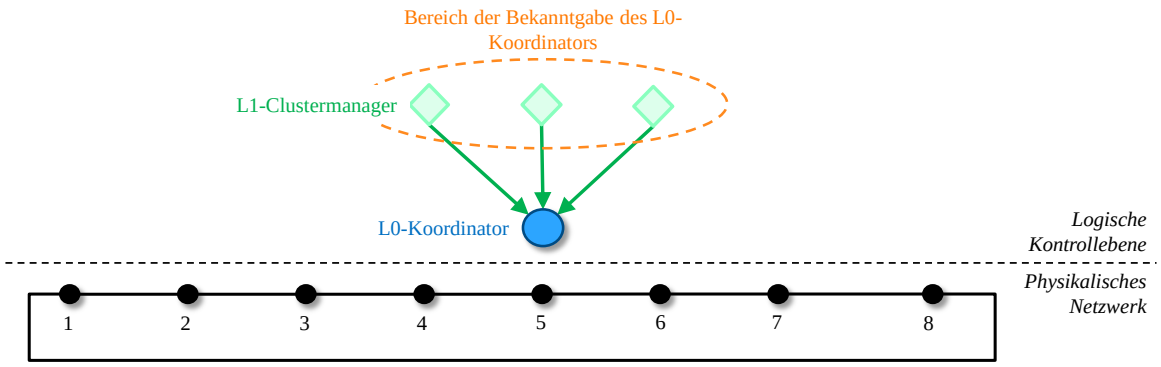

Abbildung 3.13: L1 Cluster verbinden zu bekannten L0 Koordinatoren und gliedern sie ein

In Abbildung 3.13 ist der Empfang von *RequestClusterMembership*-Nachrichten beispielhaft für einen L0-Koordinator anhand des Beispielszenarios dargestellt. Da die Ausbreitung der *AnnounceCoordinator*-Nachrichten von Hierarchielevel 0 durch den Radius *r* begrenzt ist, werden nur Clustermanager im Radius *r* eine solche Nachricht senden. Für Abbildung 3.13 sind dies die L1-Clustermanager auf den Knoten 4, 5 und 6, die sich innerhalb des gewählten Radius von 1 befinden. Als Resultat erhält der L0-Koordinator drei Kommunikationsanfragen (grüne Pfeile) von übergeordneten L1-Clustermanagern und ist nachfolgend diesen sich überlappenden übergeordneten Clustern zugeordnet. Dieser Schritt wird durch alle L1-Clustermanager für alle L0-Koordinatoren angewandt.

3.3.4.4 Instanziierung von L1-Koordinatoren

Nachdem jeder Clustermanager zu den ihm bekannten untergeordneten Koordinatoren eine Kommunikation gestartet hat, gilt es die Wahlvorgänge durchzuführen. Dadurch werden aus den überlappenden Clustern einige ausgewählt, sodass diese ausschließlich eine Koordinatorinstanz auf Level 1 erstellen. Dabei spielen (ähnlich zu Phase 1) die Prioritäten aller Wahlmitglieder eine Rolle. Innerhalb von Phase 2 zählen zu den Mitgliedern ausschließlich der jeweilige Clustermanager sowie seine untergeordneten Koordinatoren, zu denen er eine Kommunikation gestartet hat.

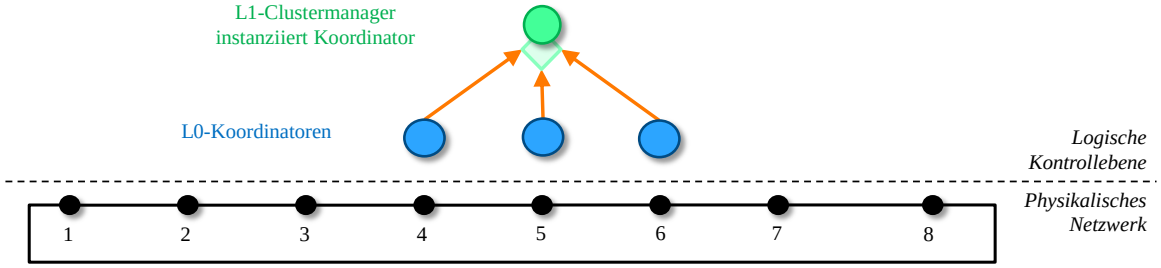

Abbildung 3.14: L1 Clustermanager empfängt Prioritäten der L0-Koordinatoren und erstellt L1-Koordinator

Abbildung 3.14 zeigt einen Ausschnitt für einen L1-Clustermanager. Er hat zuvor eine bidirektionale Kommunikation zu den drei bekannten untergeordneten Koordinatoren gestartet. Jeder erreichte Koordinator speichert lokal diese Kommunikationsinstanz ab und kann sie jederzeit für Signalisierungsnachrichten an den übergeordneten Clustermanager verwenden. Wie in Abbildung 3.14 anhand der orangenen Pfeile dargestellt, wird dies beispielsweise zum Senden der Prioritäten an den Clustermanager eingesetzt. Dieser entscheidet nachfolgend selbständig, ob er einen L1-Koordinator instanziieren muss.

Analog zu Phase 1 erstellen auch in Phase 2 nur die Clustermanager Koordinatorinstanzen, welche die höchsten Prioritäten innerhalb ihrer lokalen Nachbarschaft besitzen. In diesem Fall signalisiert ein solcher Clustermanager allen untergeordneten Wahlmitgliedern – den Koordinatoren – mit Hilfe der aufgebauten Kommunikation *Winner*-Nachrichten. Dadurch steht er als Gewinner der Wahl fest. Sollte

stattdessen ein untergeordneter Koordinator eine höhere Priorität besitzen, erstellt der Clustermanager keine Koordinatorinstanz, da er die Wahl verloren hat.

Hat der verlierende L1-Clustermanager bereits eine Koordinatorinstanz erstellt, wird diese entfernt und dieses Ereignis allen untergeordneten Wahlmitgliedern über eine *Resign*-Nachricht mitgeteilt. Folglich besitzt jeder Koordinator auf Hierarchielevel *n* stets Kenntnis darüber, welche Instanz eines Clustermanagers auf Hierarchielevel *(n + 1)* eine laufende Koordinatorinstanz besitzt. Die Abbildung 3.15 zeigt den beschriebenen Wahlvorgang verallgemeinert für einen L1+ - Clustermanager, die Abläufe sind ähnlich zu denen aus Phase 1.

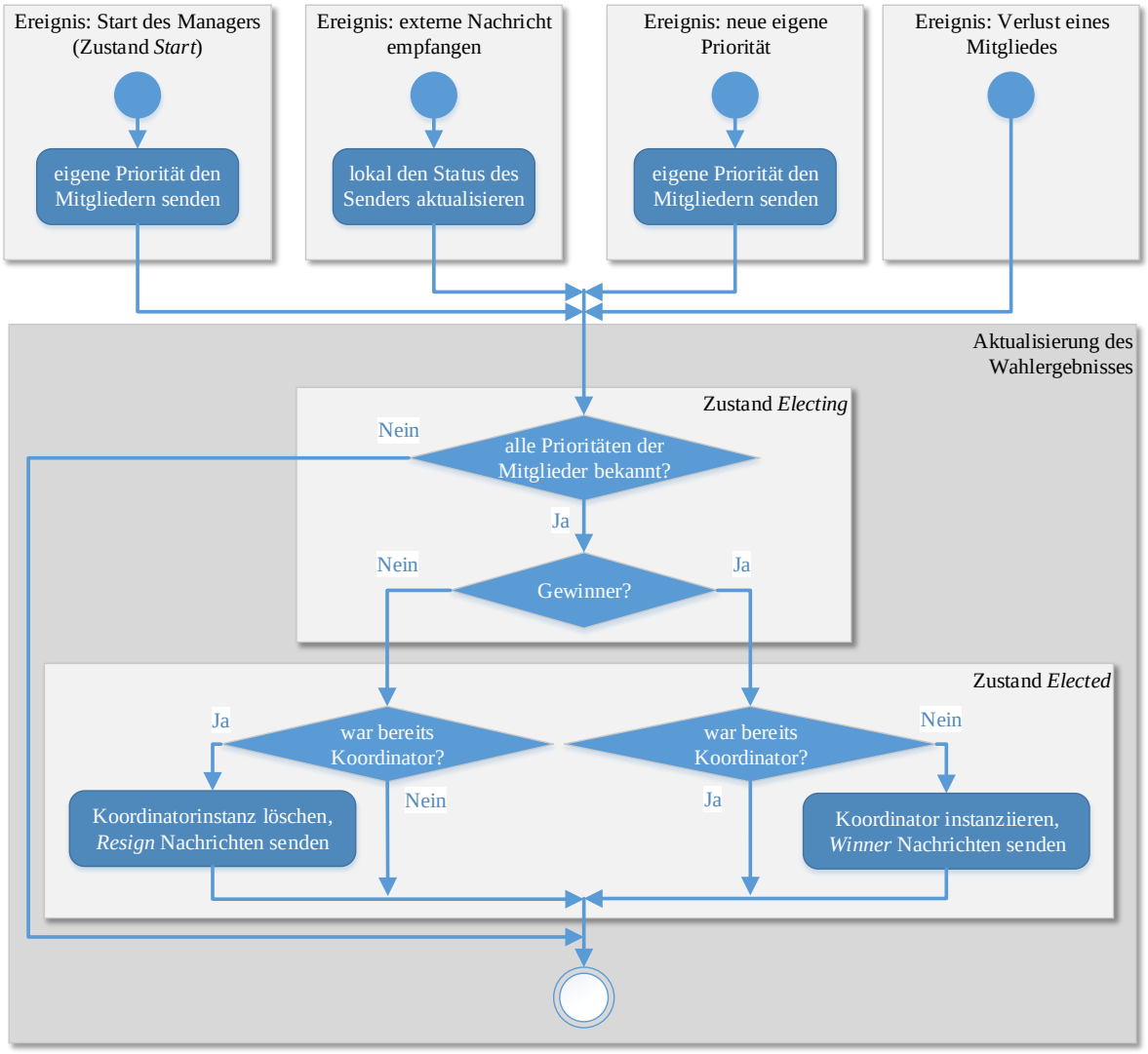

Abbildung 3.15: Reaktion eines L1+ - Clustermanagers zur Aktualisierung des Wahlergebnisses

Wie in Abbildung 3.15 zu erkennen ist, kann ein L1+ - Clustermanager erst dann ein Wahlresultat ermitteln, wenn er alle Prioritäten untergeordneter Koordinatoren kennt, zu denen er eine Kommunikation gestartet hat. Sollte ein untergeordneter Koordinator ausfallen, wird er aus dem Wahlvorgang aufgrund ausbleibender *AnnounceCoordinator*-Nachrichten als Wahlmitglied automatisch. In Abhängigkeit von den vorliegenden Prioritäten bestimmt ein Clustermanager sich selbständig als Gewinner oder Verlierer der Wahl, er wechselt anschließend vom Zustand *Electing* in den Zustand *Elected*. Der Zustand ist jedoch nicht bindend, er reagiert weiterhin auf eintreffende *PriorityUpdate*-Nachrichten von untergeordneten Koordinatoren. Sollte sich deren Priorität verändern, wechselt der Clustermanager zurück in den Zustand *Electing* und bestimmt das neue Wahlergebnis. Analog dazu verhält er sich beim Hinzukommen

oder dem Verlust von Wahlmitgliedern. Allgemein betrachtet kann ein L1+ - Clustermanager die folgenden Zustände einnehmen:

- ***Start*:** Der Clustermanager wurde erstellt und dieser signalisiert seine Priorität an untergeordneten Koordinatoren. Bisher existiert keine Kommunikation zu externen Wahlmitgliedern.
- ***Electing*:** Die Kommunikation mit untergeordneten Koordinatoren wurde gestartet. Es fehlen noch Prioritäten von Wahlmitgliedern oder es traf eine neue externe Nachricht ein. Das Wahlergebnis liegt noch nicht vor.
- ***Elected*:** Alle notwendigen Prioritäten liegen vor und das Wahlergebnis ist bestimmt, der Wahlsieger ist somit ausgewählt (Englisch: „elected"). Der Clustermanager ist entweder der Gewinner oder der Verlierer der Wahl. Im ersten Fall existiert zusätzlich eine lokale Koordinatorinstanz des jeweiligen höheren Hierarchielevels.

Fehlerfälle werden an dieser Stelle nicht näher spezifiziert. Diese sind eng mit der Implementierung gekoppelt, welche ebenfalls die Fehlerbehandlung bestimmt, sodass ein Clustermanager zurück in einen konsistenten Zustand überführt wird. Eine mögliche Variante kann dabei der Neustart des jeweiligen Clustermanagers (inklusive seiner bereits gestarteten Kommunikation zu untergeordneten Koordinatoren) sein.

3.3.4.5 Bestimmung der Prioritäten für höhere Hierarchielevels

Entscheidend für kurze Kommunikationswege – und somit ebenfalls für eine gute Gesamtperformanz des Systems – sind die je Hierarchielevel verwendeten Prioritäten eines Knotens. Es sollte stets derjenige eine Koordinatorinstanz für Level *n* erstellen, in dessen nahen Nachbarschaft sich möglichst viele Koordinatoren von Level *(n - 1)* befinden. Hierfür wird die physikalische Hopdistanz d_m zu seinen Nachbarkoordinatoren c_m verwendet, um Nachbarschaftsbeziehungen quantitativ zu bemessen.

$$p_{n,k} = \sum_{m=1}^{o_n} (-d_m)$$

Formel 3.3: Berechnung der Knotenpriorität für höhere Hierarchielevels

Formel 3.3 gibt für jeden Knoten *k* die resultierende Priorität $p_{n,k}$ auf Hierarchielevel *n* an. Sie ist abhängig von der Hopdistanz d_m von allen knotenlokal bekannten Koordinatoren c_m auf Hierarchielevel *(n - 1)*, die Werte ihrer Indizes liegen zwischen 1 und o_n. Das negative Vorzeichen invertiert den Einfluss der Distanzen, eine hohe Hopdistanz beeinflusst dadurch die resultierende Priorität entsprechend negativ. Aufgrund dessen erhalten Knoten, welche viele untergeordnete Nachbarkoordinatoren besitzen, eine höhere Priorität.

$$p_{n,k} = \sum_{m=1}^{o_n} (d_{max} - d_m)$$

Formel 3.4: Berechnung der Knotenpriorität für höhere Hierarchielevel mit Wertebereichsverschiebung

Formel 3.4 stellt den vorhergehenden Zusammenhang aus Formel 3.3 erneut dar, zusätzlich wird die maximal zulässige Hopdistanz d_{max} eingeführt, welche global als fester Parameter des Wahlalgorithmus bekannt ist. Dadurch wird der Wertebereich verschoben, sodass sich ausschließlich positive Prioritäten als Resultat ergeben. Dadurch wird ebenfalls eine Gewichtung zwischen dem Einfluss der Hopdistanzen und der Anzahl von umliegenden Nachbarkoordinatoren eingeführt, wodurch letztere einen stärkeren Einfluss auf die resultierende Priorität zugeordnet bekommt.

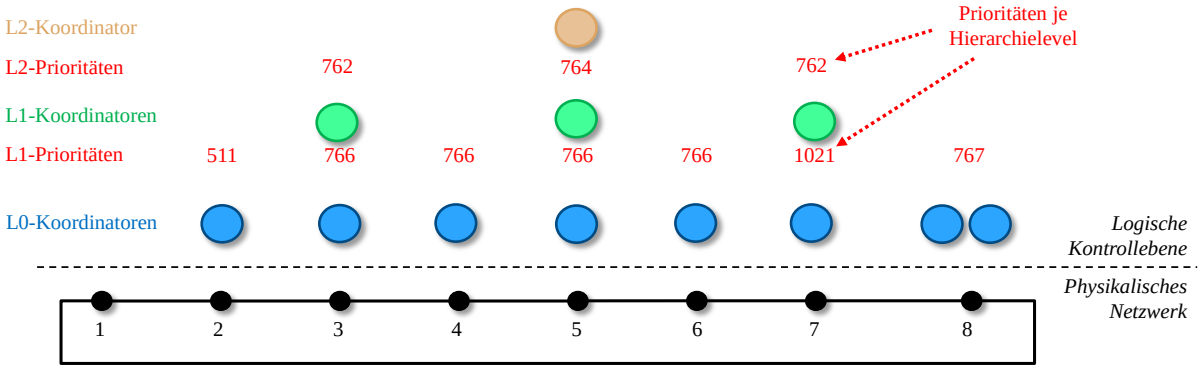

Abbildung 3.16: Knotenprioritäten auf verschiedenen Hierarchielevels und die resultierenden Koordinatorinstanzen

Nimmt man beispielsweise die maximale Hopdistanz d_{max} mit 256 an und verwendet einen Radius $r = 1$, ergibt sich für das Beispielnetzwerk die in Abbildung 3.16 dargestellte Verteilung der Prioritäten auf den höheren Hierarchielevels. Knoten 5 besitzt dabei eine L1-Priorität von 766, sie ergibt sich durch die drei lokal bekannten L0-Koordinatoren, welche sich auf den Knoten 4, 5 und 6 befinden. Dabei liegen die Knoten 4 und 6 jeweils einen physikalischen Hop entfernt, sodass der einfließende Prioritätswert jeweils 255 beträgt. Der auf Knoten 5 lokal instanziierte L0-Koordinator fließt mit einem Prioritätswert von 256 ein, sodass sich die resultierende L1-Priorität insgesamt aus 2*255+256=766 ergibt. Analog dazu werden auf Hierarchielevel 2 die Werte bestimmt, für Knoten 5 liegen in diesem Fall die umliegenden L1-Koordinatoren jeweils zwei physikalische Hops entfernt, sodass seine L2-Priorität 2*254+256=764 beträgt.

Im Netzwerk werden insgesamt drei L1-Koordinatoren instanziiert, des Weiteren befindet sich der *TOP-Koordinator* auf Knoten 5. Analog zu Phase 1 wird auch in Phase 2 die jeweilige Knoten-ID als zweites Kriterium während eines Wahlvorganges auf höheren Hierarchielevels verwendet, um stets eine totale Ordnung unter den Kandidaten zu erhalten. In Abbildung 3.16 ist dies beispielsweise an den Knoten 4 und 5 zu erkennen, der L1-Clustermanager auf Knoten 5 wurde im Vergleich zu Knoten 4 bei gleicher Priorität aufgrund der höheren Knoten-ID als Gewinner der Wahl ermittelt. Eine besondere Bedeutung hat bei diesem Wahlvorgang der L0-Koordinator auf dem Nachbarknoten 6, er liegt sowohl im Radius des L1-Clustermanagers auf Knoten 5 als auch im Radius der L1-Clustermanager von Knoten 6 und 7, Knoten 7 besitzt dabei die global höchste L1-Priorität. Der Clustermanager auf Knoten 6 kann folglich aus der Wahl nicht als Sieger hervorgehen, da er von Knoten 7 die höhere Priorität empfangen hat. Jedoch ergibt sich daraus die Frage, weshalb der L1-Clustermanager auf Knoten 5 als Wahlsieger hervorgegangen ist, während sich in seiner Nachbarschaft der L0-Koordinator auf Knoten 6 befindet und dieser bei gleicher Priorität eine höhere Knoten-ID aufweist. Um dies zu klären, bedarf es der näheren Betrachtung der sogenannten Wahlmitgliedschaften.

3.3.4.6 Deaktivierung von Wahlmitgliedschaften

Das allgemeine Ziel des Wahlalgorithmus ist es, jedem Koordinator auf Level *n* den übergeordneten Koordinator auf Level *(n + 1)* zuzuordnen, welcher im lokalen Radius *r* die höchste Priorität besitzt. Ein Koordinator kann dabei stets mehrere übergeordnete Clustermanager kennen. Somit muss er über die Möglichkeit verfügen, einem übergeordneten Clustermanager auf Level *(n + 1)* mitzuteilen, dass er Kenntnis über einen alternativen Clustermanager auf Level *(n + 1)* mit instanziiertem Koordinator besitzt, welcher eine höhere Priorität aufweist. Zu diesem Zweck wird ein Status für jede Wahlmitgliedschaft verwendet. Dieser wird im beschriebenen Fall durch den Koordinator für den übergeordneten Clustermanager mit niedrigerer Priorität deaktiviert. Um dies dem Clustermanager zu signalisieren, wird der Nachrichtentyp *Leave* eingeführt, um anzuzeigen, dass das Wahlmitglied den aktiven Status verlässt. Dadurch bezieht der empfangende Clustermanager die Priorität des sendenden Koordinators nicht mehr

in seine Wahlentscheidung ein. Die folgende Abbildung zeigt diese Signalisierung anhand des Beispielszenarios.

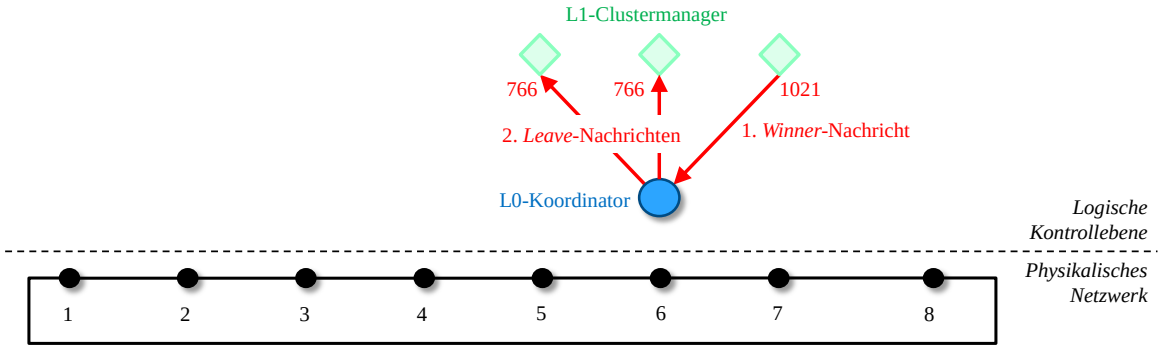

Abbildung 3.17: Deaktivierung einer Wahlmitgliedschaft als Reaktion auf eine *Winner* Nachricht

In Abbildung 3.17 ist zu erkennen, dass der L0-Koordinator auf Knoten 6 eine *Winner*-Benachrichtigung über den Wahlsieg des L1-Clustermanagers auf Knoten 7 empfängt. Als Folge sendet dieser wiederum eine *Leave*-Nachricht an die alternativen L1-Clustermanager auf den Knoten 5 und 6.

Die Wahlmitgliedschaft eines Koordinators kann zwischen verschiedenen Clustermanagern variieren, sie kann für jeden übergeordneten Clustermanager entweder aktiv oder passiv sein. Somit muss jeder Clustermanager stets eine aktuelle Liste über den Wahlstatus aller untergeordneten Koordinatoren besitzen. Sobald eine Kommunikation zwischen einem Clustermanager und einem Koordinator gestartet wird, gilt die Mitgliedschaft des untergeordneten Koordinators als aktiv und seine Priorität fließt in die Wahlentscheidung des Managers ein. Im passiven Fall wird die jeweilige Priorität ignoriert, sodass ein Clustermanager, trotz Kenntnis über einen oder mehrere untergeordnete Koordinatoren mit höherer Priorität, als Wahlgewinner hervorgehen kann. Ein Clustermanager reagiert bei Veränderung einer Wahlmitgliedschaft sofort mit erneuter Überprüfung des Wahlergebnisses.

Allgemein betrachtet werden durch einen Koordinator auf Hierarchielevel *n* immer genau dann *Leave*-Nachrichten an einen übergeordneten Clustermanager gesendet, wenn der Koordinator eine übergeordnete Koordinatorinstanz auf Hierarchielevel *(n + 1)* kennt, deren Clustermanager eine höhere Priorität aufweist. Somit sind folgende Ereignisse möglich:

- **Signalisierung durch neuen Wahlgewinner:** Von einem Clustermanager auf Hierarchielevel *(n + 1)* wird eine *Winner*-Nachricht durch einen untergeordneten Koordinator empfangen, der Clustermanager hat die Nachricht aufgrund seiner internen Abläufe nach Erstellung der neuen Koordinatorinstanz auf Hierarchielevel *(n + 1)* versandt. Die Wahlmitgliedschaft des empfangenden Koordinators ist für den sendenden übergeordneten Clustermanager aktiv.
  In diesem Fall wechselt der empfangende Koordinator mit Hilfe von *Leave*-Nachrichten den Status der Wahlmitgliedschaften für alle Clustermanager auf passiv, deren Priorität niedriger als jene des *Winner*-sendenden Clustermanagers ist.
- **Signalisierung einer Prioritätsänderung durch Wahlgewinner:** Ein Clustermanager auf Hierarchielevel *(n + 1)* mit erstellter Koordinatorinstanz signalisiert eine höhere Priorität als die bisher von ihm bekannte. Die Wahlmitgliedschaft des empfangenden Koordinators ist für den sendenden übergeordneten Clustermanager aktiv.
  In diesem Fall ist es möglich, dass weitere alternative Clustermanager auf Hierarchielevel *(n + 1)* eine niedrigere Priorität als der sendende Clustermanager besitzen. Der empfangende Koordinator deaktiviert folglich die Wahlmitgliedschaft für alle übergeordneten Clustermanager mit niedrigerer Priorität per *Leave*-Nachricht.

Folglich wird durch *Leave*-Nachrichten sichergestellt, dass jeder Koordinatorinstanz auf Hierarchielevel *n* nur die Koordinatorinstanz des übergeordneten Hierarchielevels zugeordnet ist, welche die höchste Priorität im Radius *r* besitzt. Für alle anderen Clustermanager ist die Wahlmitgliedschaft deaktiviert.

3.3.4.7 Reaktivierung von Wahlmitgliedschaften

Da sich die finalen Prioritäten auf einem höheren Hierarchielevel während der Startphase des Netzwerks erst nach einigen Signalisierungen – inklusive der damit verbundenen Verzögerungszeiten – auf dem untergeordneten Hierarchielevel ergeben, kann der Fall eintreten, dass signalisierte Prioritäten nur temporäre Werte darstellen. Sie werden durch nachfolgende Signalisierungen aktualisiert. Folglich muss es für einen Koordinator ebenfalls möglich sein, eine Mitgliedschaft erneut aktivieren zu können, was insbesondere bei Topologieänderungen im Netzwerk notwendig ist. Zu diesem Zweck werden sogenannte *Return*-Nachrichten als Gegenstück zu *Leave*-Nachrichten verwendet, welche die Signalisierung der Reaktivierung einer Wahlmitgliedschaft ermöglichen. Auf Basis der *Leave/Return*-Nachrichten stellt jeder Koordinator sicher, dass er ausschließlich für den Clustermanager ein aktives Wahlmitglied ist, welcher aktuell die Koordinatorinstanz mit der höchsten Priorität im Radius *r* aufweist.

Die Verwendung von *Return*-Nachrichten wird nachfolgend an einem Beispiel verdeutlicht. Bezogen auf das Beispielszenario ist während der Startphase des Netzwerks folgender Ablauf denkbar:

1.) **Erstellung des L1-Koordinators auf Knoten 6:** Knoten 8 könnte im Vergleich zu den anderen verzögert gestartet worden sein, sodass seine lokalen L0-Koordinatorinstanzen gegenüber den anderen im Netzwerk ebenfalls um einige Sekunden verzögert gestartet werden. Folglich ist die L1-Priorität von Knoten 7 bis zum Eintreffen der *AnnounceCoordinator*-Nachrichten von Knoten 8 geringer als ihr finaler Wert. Ist die L1-Priorität sogar geringer als jene von Knoten 6, würde dieser aufgrund der in der lokalen Nachbarschaft (begrenzt durch den Radius *r*) temporär höchsten L1-Priorität eine L1-Koordinatorinstanz erstellen. Den umliegenden L0-Koordinatoren wird dies per *Winner*-Nachricht mitgeteilt. Folglich werden diese alle Wahlmitgliedschaften bei L1-Clustermanagern mit geringerer Priorität per *Leave*-Nachricht beenden. Beispielsweise würde der L0-Koordinator von Knoten 5 die Wahlmitgliedschaften bei den L1-Clustermanagern von Knoten 4 und 5 deaktivieren.
2.) **Entfernung des L1-Koordinators auf Knoten 6:** Nimmt man als weiteren Verlauf an, dass nach vollständigem Start von Knoten 8 die L1-Priorität des L1-Clustermanagers auf Knoten 7 aufgrund der daraufhin eintreffenden *AnnounceCoordinator*-Nachrichten von Knoten 8 auf ihren finalen Wert 1021 steigt, würde ein L1-Koordinator auf Knoten 7 gestartet werden. Der L1-Clustermanager auf Knoten 6 bemerkt, dass die lokale L1-Priorität geringer als jene von Knoten 7 ist und somit der lokale L1-Clustermanager die Wahl verloren hat. Die L1-Koordinatorinstanz wurde verdrängt, sie wird entfernt und dieses Ereignis wird den untergeordneten L0-Koordinatoren über eine *Resign*-Nachricht mitgeteilt.

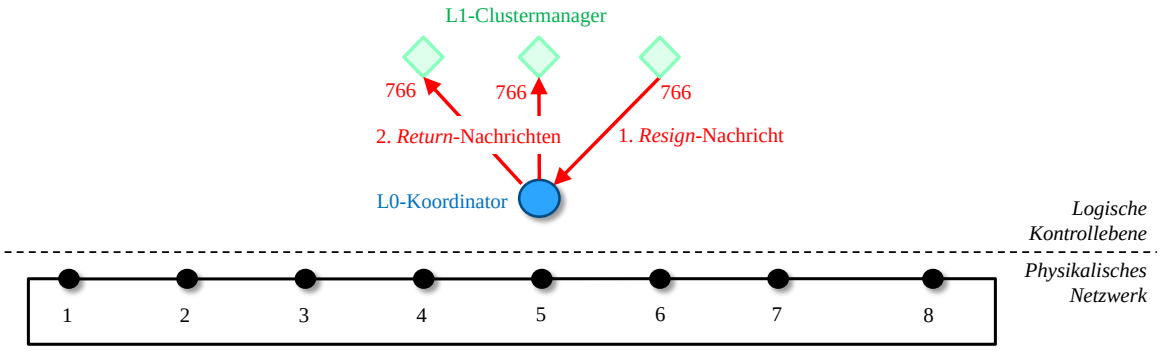

Abbildung 3.18: Reaktivierung einer Wahlmitgliedschaft als Reaktion auf eine *Resign*-Nachricht

Als Folge des beschriebenen Ablaufes ergibt sich die in Abbildung 3.18 dargestellte Reaktion des L0-Koordinators auf Knoten 5:

1.) Der L0-Koordinator erhält die zuvor beschriebene *Resign*-Nachricht vom L1-Clustermanager auf Knoten 6. Dadurch weiß der Koordinator, dass er seinen vormals übergeordneten L1-Koordinator verloren hat. Da er in dem Moment der einzige noch existierende L1-Koordinator ist, benötigt der L0-Koordinator eine neue übergeordnete Koordinatorinstanz.
2.) Im zweiten Schritt reaktiviert der L0-Koordinator auf Knoten 5 mit Hilfe von *Return*-Nachrichten seine Wahlmitgliedschaft bei den L1-Clustermanagern auf den Knoten 4 und 5. Somit kann der L1-Clustermanager auf Knoten 5 wieder als Sieger seiner Wahl hervorgehen, da der lokal untergeordnete L0-Koordinator erneut aktives Mitglied seiner Wahl ist. Zusätzlich hat der L0-Koordinator auf Knoten 6 aufgrund seiner höheren L1-Priorität seine Wahlmitgliedschaft bei ihm (dem L1-Clustermanager auf Knoten 5) deaktiviert.

Somit gewinnt der L1-Clustermanager auf Knoten 5 die Wahl und erstellt eine lokale L1-Koordinatorinstanz.

Allgemein betrachtet werden *Return*-Nachrichten immer dann versendet, wenn eines der folgenden Ereignisse eintritt[13]:

- **Neuwahl eines übergeordneten Koordinators:** Ein übergeordneter Clustermanager signalisiert über eine *Winner*-Nachricht, dass er eine neue Koordinatorinstanz erstellt hat. Diese besitzt für den empfangenden untergeordneten Koordinator die höchste Priorität im Radius *r*, sodass dieser über eine *Return*-Nachricht die zugehörige Wahlmitgliedschaft reaktiviert, insofern diese vorher inaktiv war.
- **Abwahl des letzten übergeordneten Koordinators:** Vom letzten übergeordneten Clustermanager mit gestarteter Koordinatorinstanz wird eine *Resign*-Nachricht empfangen. Sie signalisiert, dass die Koordinatorinstanz nicht länger zur Verfügung steht. Der empfangende, untergeordnete Koordinator hat somit seinen übergeordneten Koordinator mit der höchsten Priorität verloren. Als Reaktion darauf versendet er an alle übergeordnete Clustermanager eine *Return*-Nachricht, um alle Wahlmitgliedschaften zu reaktivieren.

Als Resultat der *Leave/Return*-Nachrichten wechselt der Status von Wahlmitgliedschaften solange, bis sich die Prioritäten im Netzwerk nicht mehr verändern. In diesem Fall konvergiert die Kontrollebene nach endlicher Zeit zu einem stabilen Zustand. Jeder Koordinator ist aktives Wahlmitglied des übergeordneten Koordinators mit der jeweils höchsten Priorität im Radius *r*, sodass der Algorithmus für alle Knoten terminiert. Für den beschriebenen Algorithmus wird in dieser Arbeit der Name *Distributed Coordinator Exclusion* (DCE) gewählt, der Name ist abgeleitet von seinem internen Ablauf zur gegenseitigen Verdrängung von Koordinatoren.

3.3.4.8 Ermittlung des Wahlergebnisses

Die Ermittlung eines Wahlergebnisses verwendet die Schritte von Phase 1 aus Abschnitt 3.3.2.2. Im Gegensatz dazu können in Phase 2 Wahlmitglieder ihre Mitgliedschaften deaktivieren, sodass sie nicht mehr als Kandidaten – und somit auch nicht mehr als Wahlsieger – zur Verfügung stehen. Ihre Prioritäten werden bei der Bestimmung des Wahlergebnisses ignoriert.

[13] Der Ablauf der *Leave/Return* Signalisierungen wird in Abschnitt 4.2.2 nochmals detaillierter erläutert.

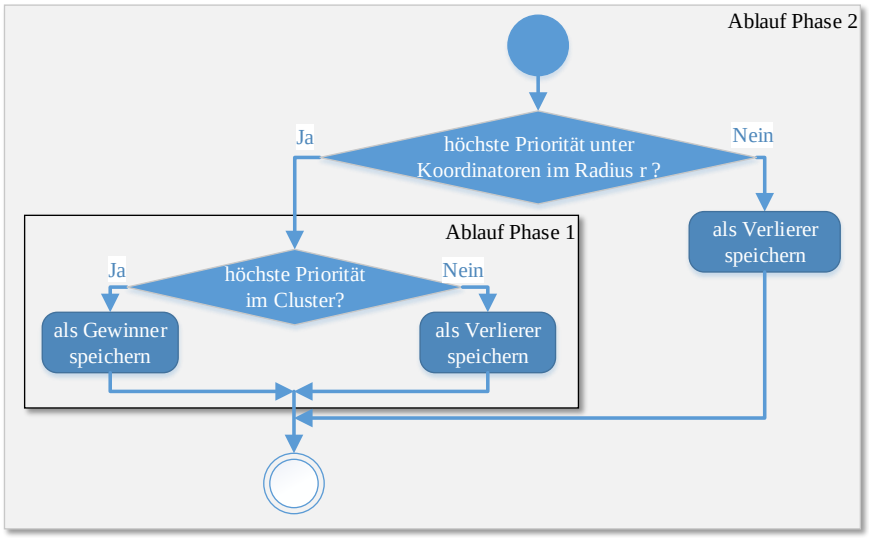

Abbildung 3.19: Gewinnerermittlung in Phase 2

Wie in Abbildung 3.19 zu sehen, übernimmt der Ablauf von Phase 2 den Ablauf von Phase 1 als Unterbedingung. Zusätzlich werden in Phase 2 die Prioritäten umliegender Koordinatorinstanzen einbezogen, welche sich innerhalb des Radius *r* befinden. Sollte auf dem jeweiligen Hierarchielevel ein umliegender Koordinator eine höhere Priorität aufweisen, geht der Clustermanager nicht als Gewinner der lokalen Wahl hervor. Dadurch verdrängt eine existierende Koordinatorinstanz stets alle im Radius *r* umliegenden mit niedriger Priorität.

3.3.4.9 Annahmen des Strukturierungsalgorithmus

Nach Erläuterung des vollständigen Algorithmus verbleibt die Frage, unter welchen Annahmen eine solche Strukturierung der oberen Hierarchielevels möglich ist. Die folgende Übersicht gibt dazu eine Antwort:

- **Korrektheit des Algorithmus:** Alle Knoten benutzen den gleichen Algorithmus mit den gleichen Konfigurationsparametern. Dazu zählen die angenommene maximale Hopdistanz d_{max} sowie der Clusterradius *r*.
- **Clustermanager kennen alle Kandidaten im Radius *r*:** Durch die Ausbreitung der *AnnounceCoordinator*-Nachrichten sind jedem Clustermanager stets alle untergeordneten Koordinatoren im Radius *r* bekannt. Die Ausnahme bildet der *TOP-Koordinator*. Im nachfolgenden Abschnitt 3.3.5 werden dazu weitere Details erläutert.
- **Kooperatives Verhalten:** Jeder existierende, bekannt gegebene Koordinator antwortet.
- **Ausschluss von Kommunikationsfehlern:** Keine der verschickten Nachrichten geht verloren oder wird auf Zwischenknoten verändert. Folglich kommen alle Nachrichten in unveränderter Form am Empfänger an.[14]
- **Konsistente Signalisierung:** Manipulierte Signalisierungen sind ausgeschlossen. Werden *AnnounceCoordinator*-Nachrichten empfangen, existiert ihr Sender tatsächlich und ist der Sender der Nachricht.
- **Erkennung ausgefallener Kandidaten:** Durch Ausbleiben von *AnnounceCoordinator*-Nachrichten wird ersichtlich, dass ein untergeordneter Koordinator auf einem entfernten Knoten ausgefallen ist. Er wird aus den Daten der umliegenden Clustermanager entfernt.
- **Totale Ordnung über Kandidaten:** Allen Knoten wird pro Hierarchielevel eine Priorität zugeordnet. Für die Wahlentscheidung eines Clustermanagers werden diese in Kombination mit

[14] Nachrichtenfehler und -ausfälle werden in Abschnitt 3.6.1 detaillierter betrachtet.

der jeweiligen Knoten-ID der Wahlmitglieder verwendet. Als Ergebnis wird stets eine eindeutige Ordnung unter den Wahlmitgliedern gefunden.

- **Determinismus des Algorithmus:** Ein Koordinator eines höheren Hierarchielevels wird unter Berücksichtigung des Status der Mitgliedschaften stets auf dem Knoten mit der für die lokale Nachbarschaft höchsten Priorität bzw. Knoten-ID instanziiert.
- **Terminierung:** Der Algorithmus benötigt für die Bestimmung des Wahlergebnisses eine endliche Anzahl Schritte.

Es ist zu erkennen, dass die Annahmen von Phase 2 sehr ähnlich zu denen von Phase 1 sind. Jedoch gilt insbesondere für die verlässliche Erkennung ausgefallener Wahlmitglieder zusätzlich die Annahme:

- **Einhaltung von angenommenen maximalen Verzögerungszeiten:** Während der Übertragung einer Nachricht wird eine obere Schranke T_{delay_E2E} für die Verzögerungszeit eingehalten. Diese Bedingung wird speziell für *AnnounceCoordinator*-Nachrichten ausgenutzt, um feststellen zu können, ob ein entfernter Koordinator ausgefallen ist. Sollte das zugrundeliegende Netzwerk teils sehr hohe Übertragungsverzögerungen aufweisen, müssen die angenommen maximalen Verzögerungszeiten entsprechend größer gewählt werden. Dies ist insbesondere zur Erkennung von Koordinatorausfällen in Abschnitt 3.3.7 wichtig.

3.3.4.10 Ergebnis von Phase 2

Für jedes höhere Hierarchielevel *n* bilden die Clustermanager mit der im Radius *r* höchsten Priorität eine Koordinatorinstanz aus. Als Folge daraus deaktivieren alle im Radius *r* befindlichen Koordinatorinstanzen auf Hierarchielevel *(n - 1)* durch *Leave*-Nachrichten ihre Wahlmitgliedschaften bei allen ihn bekannten alternativen übergeordneten Clustermanagern. Somit werden um Koordinatorinstanzen neue Cluster mit einem minimalen Abstand von *(r + 1)* logischen Hops ausgebildet.

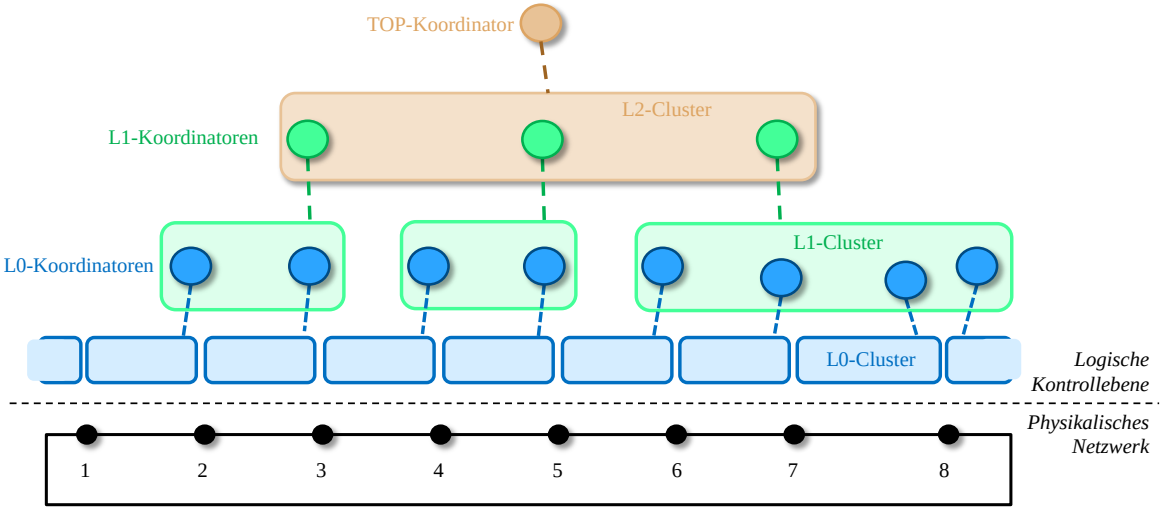

Abbildung 3.20: Resultierende Struktur der Kontrollebene

Angewandt auf das Beispielszenario ergibt sich die in Abbildung 3.20 dargestellte Struktur der Clusterverteilung innerhalb der Kontrollebene. Es ist zu sehen, dass drei Wahlsieger auf Level L1 ermittelt wurden. Sie befinden sich auf den Knoten 3, 5 und 7. Die L1-Koordinatoren sind in Abbildung 3.20 als grüne Punkte dargestellt[15]. Der Radius ist auf den Wert *r = 1* festgelegt, sodass die Distanz zwischen

[15] Zum Vergleich des Wahlergebnisses sei an dieser Stelle nochmals auf die Prioritätsverteilung aus Abbildung 27 verwiesen.

einem L1-Koordinator und seinen Clustermitgliedern maximal 1 beträgt. Des Weiteren ist in der Abbildung zu erkennen, dass auf Level 2 der *TOP-Koordinator* auf Knoten 5 instanziiert wurde. Er bildet die Spitze der Hierarchie innerhalb der Kontrollebene.

3.3.5 Ausbreitung von *AnnounceCoordinator*-Nachrichten

Die Verbreitung von Koordinatorbekanntgaben in Form von *AnnounceCoordinator*-Nachrichten ist durch verschiedene Mechanismen begrenzt. Dazu zählen ebenfalls die Sonderstatus des höchsten Hierarchielevels sowie von Endsystemen im Netzwerk.

3.3.5.1 Einhaltung des Clusterradius auf höheren Hierarchielevels

Da sich die Angabe des Clusterradius *r* stets auf das jeweilige Hierarchielevel bezieht, muss die Ausbreitung von *AnnounceCoordinator*-Nachrichten zwischen Phase 1 und 2 unterschiedlich gehandhabt werden. Für Hierarchielevel 0 aus Phase 1 genügt es, für jeden passierten physikalischen Knoten den *Hop-Zähler* innerhalb der *AnnounceCoordinator*-Nachrichten um 1 zu erhöhen. Für höhere Hierarchielevels (exklusive dem höchsten) müssen Übergänge zwischen zwei benachbarten Clusterzonen, welche einen gültigen Koordinator aufweisen, eindeutig erkannt werden, erst dann darf der *Hop-Zähler* um 1 erhöht werden. Wie in Abschnitt 3.3.5.2 näher erläutert, ist dieses Vorgehen nur für Hierarchien mit einer Tiefe größer als 3 notwendig.

Für eine korrekte Erkennung von Clusterübergängen auf höheren Hierarchielevels bei einer Tiefe ab 4 werden zwei Eingaben benötigt:

- **Bekanntgabe von übergeordneten Koordinatoren:** Jedem Knoten müssen alle IDs seiner übergeordneten Clustermanager mit gültigem Koordinator bekannt sein. Die Verteilung der hierfür notwendigen Information geschieht, ausgehend vom *TOP-Koordinator,* in Richtung des untersten Levels der Hierarchie. Jeder Koordinator benachrichtigt jede untergeordnete Entität über die ID seines ihm zugehörigen Clustermanagers. Da diese IDs während der gesamten Laufzeit eines Clustermanagers konstant bleiben, müssen die Daten jeder untergeordneten Entität nur einmalig signalisiert werden.
- **Letzter Hop in *AnnounceCoordinator*-Nachrichten:** Innerhalb von *AnnounceCoordinator*-Nachrichten muss zusätzlich die ID des Clustermanagers gespeichert sein, dessen Cluster als letztes passiert wurde. Dabei werden nur Cluster auf dem Hierarchielevel beachtet, welches dem sendenden Koordinator entspricht.

Beim Empfang einer *AnnounceCoordinator*-Nachricht eines höheren Hierarchielevels werden die Clustermanager-ID aus der Nachricht mit der ID des jeweils übergeordneten Koordinators des jeweiligen Hierarchielevels verglichen. Sollten sich beide IDs unterscheiden, muss der *Hop-Zähler* innerhalb der *AnnounceCoordinator*-Nachricht um 1 erhöht werden, bevor sie weitergeleitet wird. Dadurch beschreibt der Wert des *Hop-Zählers* stets die Anzahl bereits passierter Cluster des jeweiligen Hierarchielevels. Hat der *Hop-Zähler* bereits den Wert *r* angenommen, ist der maximal erlaubte Ausbreitungsradius erreicht und die Nachricht wird entfernt.

3.3.5.2 Sonderstatus des höchsten Hierarchielevels

Insbesondere das oberste Hierarchielevel, welches ausschließlich einen *TOP-Koordinator* instanziieren darf, benötigt eine besondere Behandlung aller *AnnounceCoordinator*-Nachrichten von untergeordneten Koordinatoren. Ihre Bekanntgabe muss allen übergeordneten Clustermanagern mitgeteilt werden, sodass diese stets alle untergeordneten Koordinatoren kennen und somit auch nur ein *TOP-Koordinator* auf dem obersten Hierarchielevel instanziiert wird. Dieser kennt wiederum alle Koordinatoren des darunterliegenden Hierarchielevels. Für eine Hierarchietiefe von 3 bedeutet das, dass die Bekanntgabe von L1-Koordinatoren nicht eingeschränkt werden darf und der TOP-Koordinator auf Hierarchielevel 2 alle L1-Koordinatoren kennt und mit ihnen Nachrichten austauscht. Würden die *AnnounceCoordinator*-

Nachrichten der L1-Koordinatoren in ihrer Ausbreitung begrenzt, würde mehr als ein TOP-Koordinator gewählt werden und die Verwaltung des Netzwerks wäre unterteilt.

3.3.5.3 Sonderstatus von Endsystemen

Insofern ein Knoten eine Konnektivität von 1 besitzt, ist er ausschließlich Mitglied einer Broadcast-Domäne und stellt somit ein Endsystem dar. Sollte es innerhalb der Domäne einen weiteren Knoten geben, der als Gateway zu anderen Netzwerken oder dem Internet dient, weist dieser eine Konnektivität größer als 1 auf. Ihm fällt die Koordinatorrolle für die Domäne zu und er instanziiert einen L0-Koordinator. Dieser wird in diesem Fall jedoch nicht den übrigen Knoten der Domäne bekanntgegeben, es gilt eine automatische Sperrzone für *AnnounceCoordinator*-Nachrichten für diese Knoten. Dadurch erhalten sie ebenfalls keine Bekanntgaben von entfernteren Koordinatorinstanzen von anderen Netzwerkabschnitten.

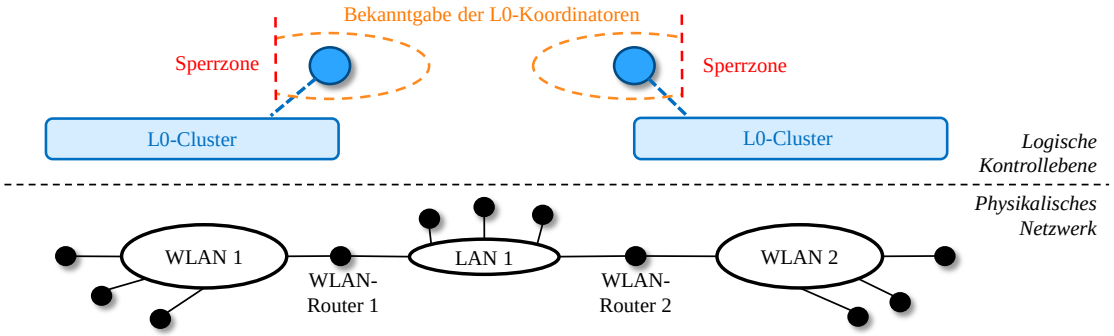

Abbildung 3.21: Sperrzonen für *AnnounceCoordinator*-Nachrichten

Abbildung 3.21 zeigt ein Beispiel für den zuvor beschriebenen Fall. Die zwei Knoten „WLAN-Router 1“ und „WLAN-Router 2“ sind jeweils Gateways eines blau markierten L0-Clusters. Sie weisen eine Konnektivität von 2 auf und haben die Wahl auf Hierarchielevel 0 gewonnen, sodass sich auf ihnen jeweils eine lokale L0-Koordinatorinstanz befindet. Je L0-Cluster existieren weitere Knoten mit einer Konnektivität von 1, sie verlieren stets eine Wahl gegenüber den Gateways und besitzen keine L0-Koordinatoren. Folglich werden sie keinesfalls einen höheren Koordinator instanziieren und benötigen somit auch keine Kenntnis über existierende L0-Koordinatoren von anderen Knoten. Sie werden somit einer Sperrzone für *AnnounceCoordinator*-Nachrichten zugeordnet.

Zur Bestimmung der Sperrzonen wird die L0-Priorität umliegender L0-Clustermanager verwendet. Sie entspricht der Konnektivität des jeweiligen Knotens. Sperrzonen werden dabei nicht explizit knotenübergreifend signalisiert, stattdessen werden auf dem jeweiligen Gateway jegliche *AnnounceCoordinator*-Nachrichten für Links zu den betroffenen Knoten blockiert. Dadurch erhält eine Broadcast-Domäne mit beliebig vielen Knoten keine Bekanntgaben, insofern sie ausschließlich einen Gateway besitzt. Sollte es mehr als ein Gateway geben, werden die verfügbaren Koordinatoren ausschließlich diesen speziellen Knoten mitgeteilt.

3.3.6 Aktualisierung von Koordinatordaten

Die Kontrollebene verwendet für das Routing von Signalisierungsnachrichten stets die kürzeste Route. In einer realen Umgebung kann es jedoch jederzeit zu spontanen Ausfällen kommen. Des Weiteren sind administrative Eingriffe in Form von Umstrukturierungen des Netzwerks denkbar. Dadurch wird die Konnektivität zwischen den Knoten im Netzwerk verändert, sodass Routen wegfallen oder neue hinzukommen. Die für einen Knoten existierende kürzeste Route kann sich somit verändern.

3.3.6.1 Einsatz eines Intervalls

Um auftretende Veränderungen an der Route zu dem Knoten eines Koordinators automatisch zu erkennen, ist es notwendig, dass die zugehörigen Routingdaten kontinuierlich gesendet werden. Dafür ist es

sinnvoll, ein Intervall für den periodischen Versand von *AnnounceCoordinator*-Nachrichten zu verwenden, sodass empfangende Knoten die lokal gespeicherten Routen automatisch aktualisieren können. Entdeckt ein Empfänger in einer *AnnounceCoordinator*-Nachricht eine kürzere Route zu einem Koordinator, wird die bisher gespeicherte Route durch diese ersetzt. Durch diesen Mechanismus lernt ein Knoten kontinuierlich die kürzeste Route zu einem Koordinator. Im Gegensatz dazu muss es auch einen Mechanismus zur Erkennung von ungültig gewordenen Routen geben. Zu diesem Zweck besitzt die lokal gespeicherte Route generell eine begrenzte Gültigkeitsdauer und muss kontinuierlich bestätigt werden. Bleibt diese Bestätigung jedoch für eine definierte Zeitdauer aus, gilt die Route als ungültig und wird durch die zuletzt als gültig empfangene ersetzt. Auf Basis der nachfolgend eintreffenden Aktualisierungen lernt der Knoten wiederum die neue kürzeste Route.

3.3.6.2 Dynamik des Intervalls

Würde man *AnnounceCoordinator*-Nachrichten stets mit konstantem Intervall versenden, bleibt das dadurch verursachte Signalisierungsaufkommen ebenfalls konstant. Sinnvoller ist ein dynamisches Intervall, dessen Wertebereiche sich aus den folgenden Phasen der Hierarchieerstellung ergeben:

- **Instabile Hierarchie:** Die Route zu dem Knoten eines Koordinators verändert sich insbesondere zur Startphase des Netzwerks sowie bei Veränderungen der Konnektivität im Netzwerk. In dieser Zeit entscheidet es sich, ob der Koordinator bestehen bleibt oder wiederum durch den Wahlalgorithmus durch einen alternativen ersetzt wird.
- **Stabile Hierarchie:** Insofern ein Koordinator eine definierte Lebenszeit $T_{stable\_hierarchie}$ bereits überschritten hat, kann von einer stabilen Platzierung des Koordinators ausgegangen werden. In dem Fall ist es sinnvoll, die Senderate für *AnnounceCoordinator*-Nachrichten zu reduzieren.

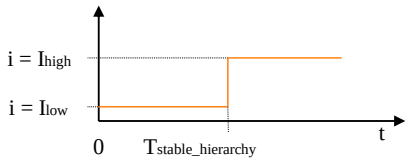

Abbildung 3.22: Intervalle von *AnnounceCoordinator*-Nachrichten

Abbildung 3.22 zeigt die resultierenden Aktualisierungsintervalle für *AnnounceCoordinator*-Nachrichten in Abhängigkeit von der Lebenszeit des jeweiligen Koordinators. Zu Beginn wird das niedrige Intervall I_{low} verwendet, sodass die Aktualisierungsrate hoch ist. Die Konvergenzzeit zur Erstellung einer stabilen Hierarchie wird dadurch niedrig gehalten. Erst nach Ablauf der Zeit $T_{stable\_hierarchy}$ gilt die Existenz des Koordinators als bestätigt – die Hierarchie wird bezüglich des Koordinators als stabil angenommen. Ab diesem Zeitpunkt wird das Intervall auf den Wert I_{high} erhöht, sodass die Aktualisierungen und somit auch die verursachte Datenrate reduziert werden. Dennoch sollten die Aktualisierungsnachrichten in dem Fall dennoch häufig genug versendet werden, um es der Kontrollebene weiterhin zu ermöglichen, in akzeptabler Zeit auf Topologieveränderungen zu reagieren. Ist beispielsweise eine erste Reaktion nach spätestens 1 Minute notwendig, sollte das Intervall diesen Wert nicht überschreiten, da ansonsten die Erkennung von Ausfällen zu lange dauert. Weitere Details zur Alterung von Koordinatordaten und der Erkennung von Ausfällen werden in den nachfolgenden beiden Abschnitten 3.3.7 und 3.3.8 gegeben.

3.3.7 Knotenausfall und Entfernung von Koordinatorinstanzen

Da in Netzwerken durch Hardwaredefekte spontan Knoten ausfallen können, kann eine Koordinatorinstanz ebenfalls plötzlich nicht mehr zur Verfügung stehen. In einem solchen Fall kann der ausfallende Koordinator keine explizite Signalisierung zur Meldung dieses Ereignisses an andere Knoten ausführen.

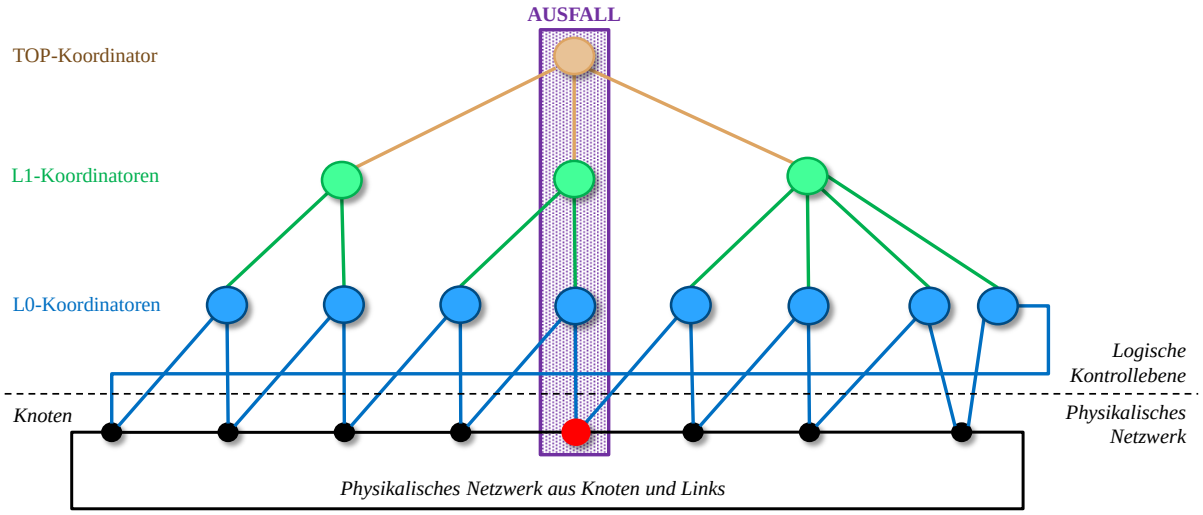

Abbildung 3.23: Ein Knotenausfall kann zu Ausfällen auf allen Hierarchielevels führen

Ein Ausfall eines Knotens kann sich im schlimmsten Fall auf alle Level der Hierarchie auswirken. Dieser Fall ist in Abbildung 3.23 dargestellt. Es ist erkennbar, dass der Ausfall des rot markierten Knotens im Beispielnetzwerk zum Verlust von jeweils einem L0- und L1-Koordinator führt. Zusätzlich ist der *TOP-Koordinator* betroffen, der aufgrund des Knotendefektes nicht mehr als Kopf der Hierarchie verwendet werden kann. Die Kontrollebene muss in der Lage sein, Ausfälle von Koordinatoren zu erkennen und automatisch zu beheben, um der Anforderung nach autonomer Arbeitsweise gerecht zu werden.

Ein erster Lösungsansatz ist die Verwendung von zusätzlichen Hallo-Paketen, welche zum periodischen Testen auf Erreichbarkeit und Funktion für jeden Koordinator verwendet werden. Bei jedem Test benötigt dieses Vorgehen ein zusätzliches Anfrage- und Antwortpaket. Daraus ergeben sich zusätzlicher Datenaufwand sowie Signalisierungskomplexität im Netzwerk. Der für HRM verwendete Lösungsansatz wird im Folgenden detaillierter erläutert.

3.3.7.1 Alterung von Koordinatordaten

Statt zusätzlicher Hallo-Pakete verwendet HRM die periodischen *AnnounceCoordinator*-Nachrichten aus dem vorherigen Abschnitt 3.3.6, um den Ausfall eines Koordinators zu erkennen. Diese Nachrichten werden vom jeweiligen Koordinator durch Fluten eines begrenzten Netzwerkbereiches verteilt. Als Vorteil dieses Vorgehens steht der geringere Signalisierungsaufwand im Vordergrund. Um ausgefallene Koordinatoren zu erkennen, wird den Daten über existierende Koordinatoren eine Lebenszeit zugewiesen. Sie wird durch eine Gültigkeitsdauer festgelegt, welche in den *AnnounceCoordinator*-Nachrichten dem Empfänger mitgeteilt wird. Ist diese Zeitdauer vergangen, ohne eine weitere Nachricht zu empfangen, wird der Koordinator durch alle Empfänger der jeweiligen Nachrichten als defekt angenommen.

Zur Beschreibung der Gültigkeitsdauer von Koordinatoren müssen absolute Zeiten vermieden werden, da eine knotenübergreifende Uhrensynchronisation nur über zusätzliche Signalisierungen (bspw. das NTP-Protokoll [105]) erreicht werden kann. Für HRM werden stattdessen relative Zeitangaben innerhalb der *AnnounceCoordinator*-Nachrichten eingesetzt.

$$t_{invalidate\_coordinator} = t_{receive\_time} + t_{coordinator\_acknowledge} + T_{delay\_E2E}$$

Formel 3.5: Berechnung der Zeit für die Gültigkeit von empfangenen Koordinatordaten

Der Empfänger kann auf Basis der übermittelten Zeitdauer $t_{coordinator\_acknowledge}$ den absoluten Zeitpunkt $t_{invalid\_coordinator}$ für die Annahme eines ungültig gewordenen Koordinators nach Formel 3.5 berechnen. Für die drei Eingabegrößen der Formel gilt:

- Die Zeit $t_{receive\_time}$ gibt die jeweilige lokale Uhrzeit des Empfängers der Nachricht an. Der Wert der lokalen Uhrzeit muss dabei mit korrekter und konstanter Geschwindigkeit monoton wachsen.
- Der vom Sender übermittelte Wert für $t_{coordinator\_acknowledge}$ gibt die Zeitdauer für die Bestätigung eines Koordinators an und muss stets größer als das verwendete Aktualisierungsintervall *i* aus Abschnitt 3.3.6 sein. Eine Aktualisierungsnachricht – und somit auch der Koordinator – müssen mindestens die Zeit zwischen zwei aufeinanderfolgenden Aktualisierungsnachrichten gültig bleiben. Der jeweilige Wert von $t_{coordinator\_acknowledge}$ ist implementierungsspezifisch[16].
- Zusätzlich werden die Signallaufzeiten im Netzwerk einbezogen. Sie werden durch den konstanten Wert T_{delay_E2E} beschrieben. Er gibt die maximal zu erwartende Ende-zu-Ende-Verzögerung während der Übertragung einer *AnnounceCoordinator*-Nachricht an.

Erst wenn die lokale Uhrzeit einen Wert größer oder gleich der berechneten Zeit $t_{invalidate\_coordinator}$ annimmt, muss der Koordinator als ungültig markiert werden. Die Kontrollebene reagiert in dem Fall entsprechend Abschnitt 3.3.4 durch folgende Schritte:

- **Aktualisierung der Prioritäten:** Davon betroffen sind jene Knoten, welche sich im Radius des Koordinators befinden. Auf diesen wird die lokale Priorität entsprechend verringert.
- **Aktualisierung der Clusterbildung:** Auf den Knoten, welche zum Radius des Koordinators gehören, werden die lokalen Daten aktualisiert. Dabei wird der ungültig gewordene Koordinator für den jeweils lokal gespeicherten übergeordneten Clustermanager als ungültig markiert und die laufende Kommunikation gestoppt. Zuletzt werden die vorhandenen Statusdaten über den Koordinator entfernt und er wird nicht länger als Clustermitglied geführt.

Aufgrund der veränderten Prioritäten und Clustermitgliedschaften können sich nachfolgend weiteren Veränderungen in der Hierarchiestruktur auf Nachbarknoten entsprechend Abschnitt 3.3.4 ergeben.

3.3.7.2 Gültigkeitsdauer einer Kommunikation

Die Alterung von Koordinatordaten wird für übergeordnete Clustermanager zur Erkennung von ausgefallenen Koordinatoren verwendet, zu denen bereits eine Kommunikation besteht. Zusätzlich kann der Totalausfall eines Knotens weitere Kommunikationen betreffen, deren Endpunkt sich ebenfalls auf dem Knoten des ungültig gewordenen Koordinators befindet. Somit können auch weitere Entitäten der Kontrollebene betroffen sein. Um den Status jeder Kommunikation stets aktuell zu halten, ist ein weiterer Mechanismus notwendig:

1. **Gültigkeitsdauer einer Kommunikation:** Jeder Knoten legt im Falle eines Koordinatorausfalls lokal eine Gültigkeitsdauer für jede Kommunikationsinstanz fest, deren anderer Endpunkt sich auch auf dem Knoten des ausfallenden Koordinators befindet. Zur Bestimmung der betroffenen Instanzen werden die Knoten-IDs der Gegenstelle verwendet, welche entsprechend Abschnitt 3.3.3 für jede Instanz bekannt sind.
2. **Stopp der Kommunikation:** Sollte innerhalb der Gültigkeitsdauer kein weiteres Paket eintreffen, wird die jeweilige Kommunikation und die dahinter befindliche Entität ebenfalls als ungültig angenommen. Dies kann sowohl ein entfernter Koordinator als auch ein Clustermanager sein. Die Kommunikation wird gestoppt und betroffene lokale Statusdaten werden entfernt.

[16] Die innerhalb der Implementierung verwendeten Werte sind in Abschnitt 6.3 zu finden.

Sollten aufgrund von Laufzeitverzögerungen im Netzwerk einige *AnnounceCoordinator*-Nachrichten nach Ablauf der Gültigkeitsdauer eintreffen, reagiert die Kontrollebene entsprechend Abschnitt 3.3.4 und die Kommunikation wird erneut gestartet[17].

3.3.8 Ausfall von Links

Ein physikalischer Link wird stets von Kommunikationskanälen verschiedener Hierarchielevels genutzt. Folglich kann ein Ausfall die Kommunikation auf allen Hierarchielevels betreffen.

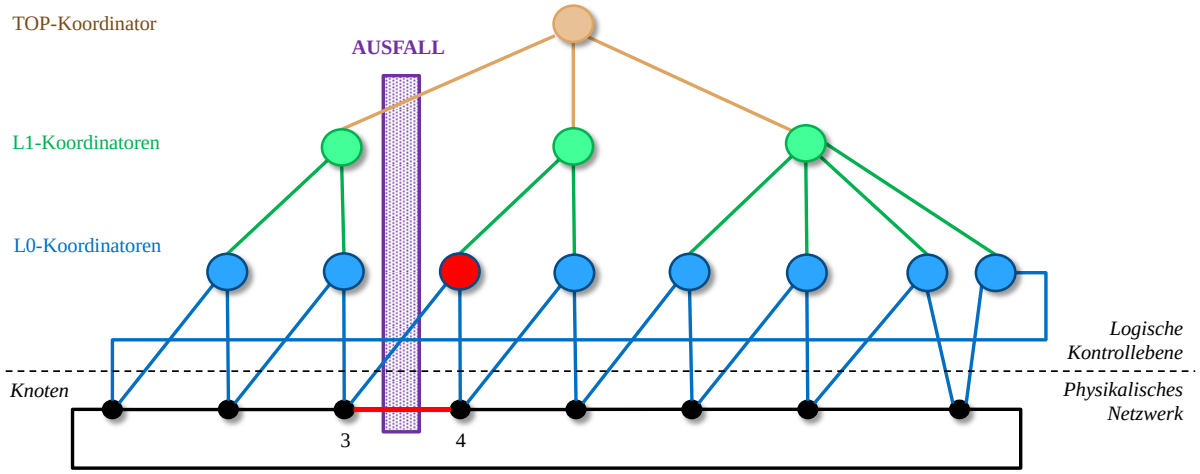

Abbildung 3.24: Ein Linkausfall führt zum Kommunikationsausfall auf verschiedenen Hierarchielevels

Abbildung 3.24 zeigt für das Beispielnetzwerk einen Ausfall des rot markierten Links zwischen Knoten 3 und 4. Dieser wirkt sich auf die Kommunikation von Hierarchielevel 0 und 2 aus. Die Kontrollebene muss solche Kommunikationsausfälle automatisch erkennen und kompensieren. Dabei basiert die Erkennung auf den periodischen Abläufen von Abschnitt 3.3.1 in Form der *AnnounceNeighborNode*-Nachrichten. Sollte ein Link zu einem Nachbarn ausfallen, erkennt dies die Kontrollebene durch Ausfall der Netzwerkschnittstelle bzw. durch Ausbleiben von Antworten auf lokal versandte *AnnounceNeighborNode*-Nachrichten. Der für den rot markierten Link zuständige L0-Koordinator ist in Abbildung 3.24 ebenfalls rot markiert. Er entfernt den ehemals erreichten Clustermanager des Nachbarknotens 3 aus seinen lokalen Daten. Dadurch ist kein weiteres Clustermitglied übrig, der L0-Cluster und sein Koordinator werden folglich entfernt. Als Resultat bleiben seine *AnnounceCoordinator*-Nachrichten auf entfernten Koordinatoren aus. Dadurch werden ebenfalls die Koordinatordaten auf umliegenden Knoten aktualisiert. Es ergeben sich in diesem Fall auf Hierarchielevel 1 neue Prioritäten für die umliegenden Knoten, welche innerhalb des Radius des entfernten Koordinators liegen. Diese werden mit Hilfe von *PriorityUpdate*-Nachrichten an die jeweils umliegenden Knoten verteilt. Innerhalb der Hierarchie können somit weitere Veränderungen auftreten.

3.3.9 Anforderungen an den Netzwerkstack

Betrachtet man die Signalisierungen zur Instanziierung der Kontrollebene aus globaler Sicht, werden zwei benötigte Basisfunktionen eines untergeordneten Protokolls ersichtlich:

- **Erkennung von Nachbarknoten:** Das untergeordnete Protokoll muss eine Schnittstelle bieten, um unbekannte Nachbarknoten ermitteln zu können. Dies wird insbesondere für *AnnounceNeighborNode*-Nachrichten aus Abschnitt 3.3.1 benötigt. Dabei kann eine spezielle Adresse zur Adressierung aller Nachbarknoten zum Einsatz kommen, ein Beispiel dafür ist die Broadcast-Adresse für *Ethernet Frames*. Alternativ kann die Erkennung von Nachbarknoten

[17] Weitere implementierungsspezifische Mechanismen zur schnellen Erkennung von Ausfällen einzelner Kommunikationspartner werden in Abschnitt 4.2.2 erläutert.

automatisch durch das untergeordnete Protokoll erfolgen, dieser Ansatz wird beispielsweise innerhalb der Implementierung verwendet (siehe Abschnitt 4.2.1).

- **Identifikation von Nachbarknoten:** Um Nachrichten von einem Knoten zum nächsten Knoten zu übertragen, muss ein Nachbarknoten mit Hilfe des untergeordneten Protokolls eindeutig identifizierbar sein. Dadurch muss sichergestellt sein, dass die Nachricht innerhalb einer Broadcast-Domäne dem gewünschten Zielknoten zugestellt wird.

3.3.10 Vergleich der Algorithmen von Phase 1 und 2

Die nachfolgende Tabelle gibt zusammenfassend einen Überblick über die in Phase 1 und 2 beschriebenen Algorithmen zur Platzierung von Koordinatorinstanzen.

	Phase 1	**Phase 2**
Clusterbildung	je Broadcast-Domäne (Cluster beinhaltet alle Knoten der Domäne)	agglomarativ je Clustermanager (Cluster beinhaltet untergeordnete Koordinatoren im Radius *r*)
Koordinator-instanzen	je Broadcast-Domäne	je ausgewähltem Clustermanager (DCE-Algorithmus verdrängt Instanzen)
Wahlmitglieder kennen einander	ja	nein (ein Clustermanager kennt nur alle untergeordneten Koordinatoren im Radius *r*, aber nicht alle alternativen Clustermanager im Radius *r* kennen einander)
Koordinator-platzierung	Anpassung des Bully-Algorithmus: • variable Prioritäten (Konnektivität und Knoten-ID) • neue Nachrichtentypen: *PriorityUpdate*, *Resign* • keine *Elect/Reply*-Nachrichten	neuer Algorithmus: • variable Prioritäten (physikalische Distanzen zu Koordinatoren des jeweils untergeordneten Hierarchielevels) • gegenseitige Verdrängung von Koordinator-instanzen durch DCE-Algorithmus (*Leave/Return*-Nachrichten)
Nachrichtentypen	*PriorityUpdate*, *Winner*, *Resign*	*PriorityUpdate*, *Winner*, *Resign*, *Leave*, *Return*
Unterstützung von Topologiedynamik	ja (*PriorityUpdate*-Nachricht)	ja (*PriorityUpdate*-Nachricht)
Beachtung von Latenzen	ja (Ausfallerkennung benutzt Konstante T_{delay_E2E})	ja (Ausfallerkennung benutzt Konstante T_{delay_E2E})
Konfigurationsparameter	keine (Radius *r* wird in Phase 1 nur für die Begrenzung der *AnnounceCoordinator*-Nachrichten verwendet)	maximaler Radius *r*, maximale Hopdistanz d_{max}

Tabelle 3.2: Vergleich zwischen den Wahlalgorithmen aus Phase 1 und 2

Wie aus Tabelle 3.2 ersichtlich, wird für Phase 1 der Bully-Algorithmus in angepasster Form verwendet, während in Phase 2 der neu entwickelte DCE-Algorithmus zum Einsatz kommt. Somit unterscheiden sich beide Phasen grundlegend voneinander.

3.4 Protokoll zur Adresszuweisung

Nachdem die hierarchische Kontrollebene von HRM vollständig aufgebaut ist und der *TOP-Koordinator* feststeht, kann das Protokoll zur Adresszuweisung starten. Dabei hat die resultierende Verteilung einen direkten Einfluss auf die Gesamtperformanz des Routingmanagements. Werden Adressen entsprechend der physikalischen Netztopologie gruppiert vergeben, ist es möglich, die Route zu einer Kno-

tengruppe über einen einzigen Eintrag einer Routingtabelle zu beschreiben. Wie in Abschnitt 2.1.8 erläutert, wird dieses Vorgehen bereits heute zur Zielaggregation in Routingtabellen angewandt. Bei HRM erfolgt die Adressvergabe ebenfalls auf Basis von Gruppen, die Kontrollebene gibt dafür bereits in Abhängigkeit von der physikalischen Topologie eine entsprechende Unterteilung der Knoten vor. Als Ergebnis besitzt jede Netzwerkschnittstelle eines Knotens eine eindeutige „HRM-Identifikation“ (kurz: „HRMID“), über die der zugehörige Knoten für andere eindeutig als Ziel identifizierbar ist. Dies entspricht dem Vorgehen in heutigen IP-basierten Netzwerken, in welchen jede Netzwerkschnittstelle ebenfalls ihre eigene IP-Adresse besitzt und diese für das angeschlossene Netzwerk einzigartig ist.

3.4.1 Adressierungsschema

Eine HRMID besteht allgemein aus einer Sequenz von Nummern, sodass der Aufbau von HRMIDs dem von *Hierarchical Location Identifiers* (HLI) [106] entspricht. Im Gegensatz zu der dortigen Annahme einer manuellen Konfiguration werden im Kontext von HRM die Nummern automatisch je Hierarchielevel vergeben. Eine HRMID besteht somit für eine Kontrollebene mit *n* Hierarchielevels aus *n* Nummern, welche durch Punkte voneinander getrennt notiert werden. HRMIDs werden verwendet für:

- **Netzwerkschnittstellen:** Der Zielknoten von Paketen einer Anwendung wird bei HRM mit Hilfe einer HRMID festgelegt. Sie identifiziert eindeutig eine Netzwerkschnittstelle des Zielknotens.
- **Koordinatoren:** Jeder Koordinator besitzt eine zugewiesene HRMID. Die einzige Ausnahme stellt dabei der TOP-Koordinator an der Spitze der Hierarchie dar: er besitzt keine HRMID.
- **Cluster:** Mehrere Knoten können mit Hilfe einer Clusteradresse aggregiert und über die zugehörige HRMID identifiziert werden. Dies spielt im weiteren Verlauf dieses Kapitels insbesondere im Kontext der Verteilung von Routingdaten und der Speicherung von Routingtabellen eine wichtige Rolle.

HRMID	Bedeutung
3.0.0	L1-Koordinator oder L1-Cluster
3.2.0	L0-Koordinator oder L0-Cluster
3.2.2	Netzwerkschnittstelle eines Knotens

Tabelle 3.3: Beispiele für die Benutzung von HRMIDs bei einer Hierarchietiefe von 3

In Tabelle 3.3 sind ausgewählte Beispiele für die Verwendung von HRMIDs gegeben. Dabei wird ersichtlich, dass ein Cluster und sein zugehöriger Koordinator stets die gleiche Adresse besitzen, welche als letzte Nummer immer eine Null aufweist. Die Unterscheidung zwischen einem Cluster und seinem Koordinator ergibt sich jeweils aus dem Signalisierungskontext. Als letztes Beispiel zeigt die Tabelle die HRMID einer Netzwerkschnittstelle. Sie besteht ausschließlich aus Nummern, welche ungleich Null sind.

3.4.2 Adressvergabe

Eine Adressvergabe, ähnlich DHCP, wäre für HRM ebenfalls denkbar, sie aber aufgrund der dafür notwendigen zentralen Verwaltungsinstanz nachteilig. Des Weiteren erfordert dies eine konfigurierte Zuordnung zwischen Knoten und Adresse. Im Gegensatz dazu werden bei HRM die Adressen durch die hierarchische Kontrollebene mit Hilfe der verteilt platzierten Koordinatoren vergeben. Zur Signalisierung werden die sogenannten *AssignHRMID*-Nachrichten verwendet.

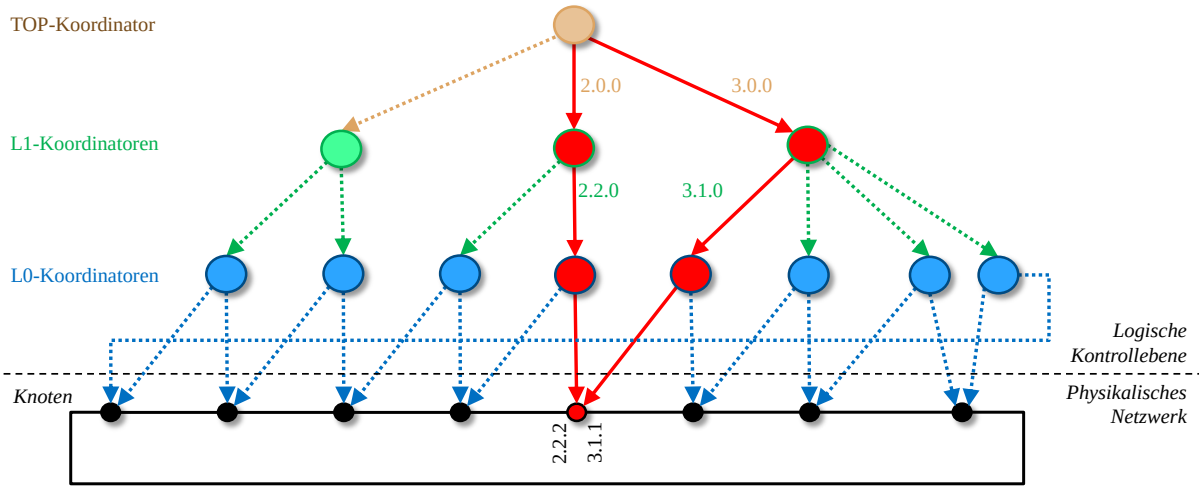

Abbildung 3.25: Hierarchische Adresszuweisung von oben nach unten durch die Kontrollebene

Abbildung 3.25 stellt die Signalisierungen zur Adresszuweisung als Ganzes für das bereits in vorherigen Erklärungen verwendete Beispielszenario dar. Beginnend am *TOP-Koordinator* und an den Blättern der Hierarchie endend werden die Adressen abwärts der Hierarchie zugewiesen. Als Ergebnis dieses Vorganges auf allen Hierarchielevels ist jeder Netzwerkschnittstelle eines Knotens eine eigene HRMID zugewiesen, über die der zugehörige Knoten im Netzwerk erreichbar ist. Die rot markierte Kommunikation in Abbildung 3.25 stellt ausgewählte Adresszuweisungen dar, welche für den rot markierten physikalischen Knoten 5 notwendig sind. Über die Koordinatoren 2.0.0/2.2.0 und 3.0.0./3.1.0 werden die HRMIDs 2.2.2 und 3.1.1 für seine beiden Netzwerkschnittstellen signalisiert. Die gestrichelten Linien zeigen weitere parallel ablaufende Signalisierungen der Adresszuweisung.

3.4.3 Adresszuweisungsprozess

Ein Koordinator kann eine Adresszuweisung nur beginnen, wenn er entweder der *TOP-Koordinator* ist oder bereits seine eigene HRMID von seinem übergeordneten Koordinator erhalten hat.

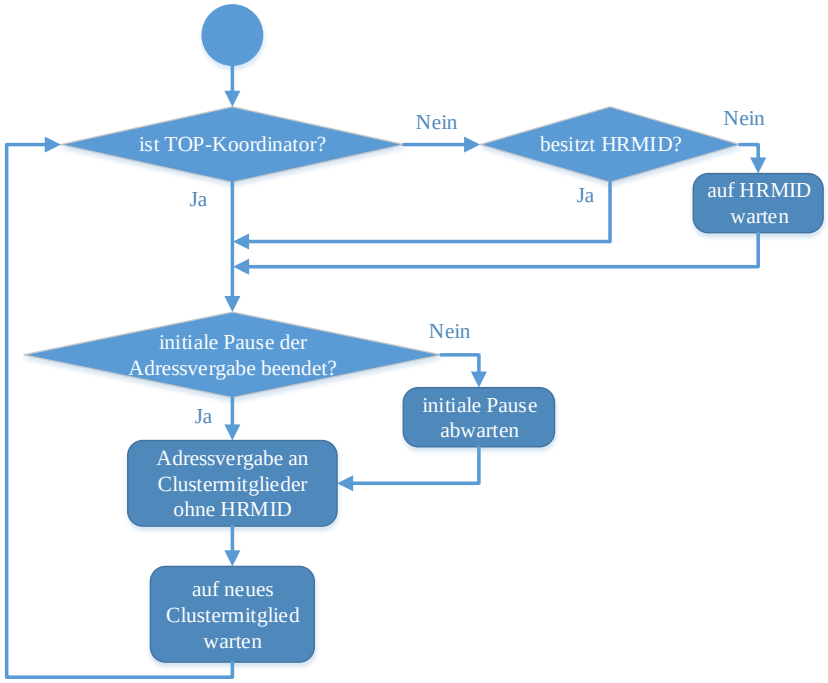

Abbildung 3.26: Ablauf des Prozesses zur Adresszuweisung

Abbildung 3.26 zeigt den resultierenden Ablauf der Adresszuweisung. Es ist zu erkennen, dass die Adressvergabe für jedes bekannte Clustermitglied erst durchgeführt wird, wenn der Koordinator eine

eigene HRMID besitzt oder er der *TOP-Koordinator* ist. Des Weiteren ist es sinnvoll, die Adresszuweisung durch einen Koordinator erst zu starten, wenn seine Instanziierung als stabil angenommen werden kann. Dies wird über eine initiale Wartezeit realisiert, wodurch temporär gültige Adresszuweisungen, beispielsweise bei Umstrukturierungen der Kontrollebene, vermieden werden. Eine zu lange Wartezeit kann dabei jedoch die Konvergenzzeit bei Topologieveränderungen negativ beeinflussen. Der genaue Wert ist implementierungsabhängig, ein typischer Wert ist in Abschnitt 6.2.2 zu finden.

Kommen während der Laufzeit eines Koordinators zusätzliche Clustermitglieder hinzu, erhält jedes unter den zuvor genannten Bedingungen ebenfalls eine Adresse zugewiesen. Die resultierenden Zuweisungen innerhalb des Clusters speichert ein Koordinator lokal ab, bei Wegfall eines Clustermitgliedes werden diese Daten entsprechend aktualisiert. Ein Koordinator kennt somit stets die noch frei verfügbaren Adressen und eine doppelte Vergabe von Adressen innerhalb eines Clusters ist ausgeschlossen.

3.4.4 Stabilisierung der Adressverteilung

Veränderungen in der Topologie können zu Umstrukturierungen in der Kontrollebene führen. Als Folge kann sowohl die Platzierung der L0-Koordinatoren und des *TOP-Koordinators* betroffen sein.

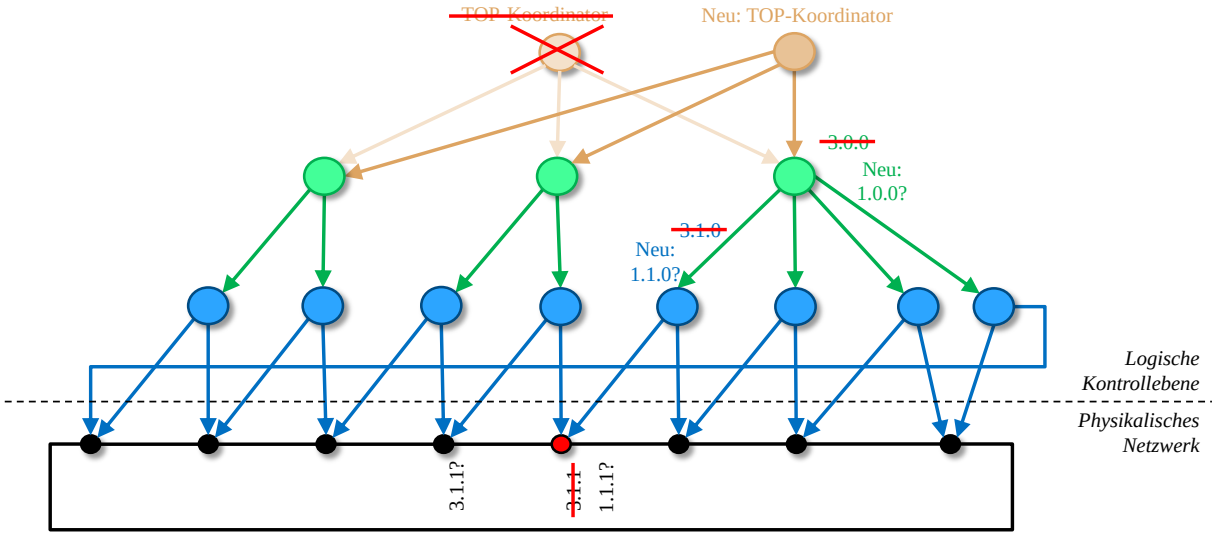

Abbildung 3.27: Veränderte Adresszuweisungen bei Hierarchieveränderungen

Ohne weitere Vorkehrungen kann eine Neuplatzierung des *TOP-Koordinators*, wie in Abbildung 3.27 dargestellt, zu sofortiger Adressneuvergabe im gesamten Netzwerk führen. Dieser Fall tritt ein, wenn der neue *TOP-Koordinator* im Vergleich zum vorherigen die untergeordneten Clustermitglieder mit neuer Reihenfolge speichert und in dieser auch die Adressvergabe durchführt. Infolgedessen werden in Abbildung 3.27 sowohl den grün dargestellten L1-Koordinatoren als auch den blau dargestellten L0-Koordinatoren neue Adressen zugewiesen. Netzwerkschnittstellen einzelner Knoten erhalten dadurch ebenfalls neue Adressen. Dies wird an Adresse 3.1.1 deutlich, welche statt dem rot markierten nachfolgend einem anderen Knoten zugeordnet ist.

Durch eine Umadressierung wird ebenfalls das Routing negativ beeinflusst. Versendet eine Anwendung im Netzwerk aus Abbildung 3.27 ihre Daten stetig an Adresse 3.1.1, werden diese nach erfolgter Umstrukturierung zum falschen Knoten geroutet. Um eine zuverlässige Übertragung von Daten im Netzwerk zu gewährleisten, müssen sowohl das Routing als auch die Adressvergabe möglichst stabil gehalten werden. Zu diesem Zweck wird eine zusätzliche Signalisierung zwischen einem Koordinator und seinem übergeordneten Koordinator eingeführt.

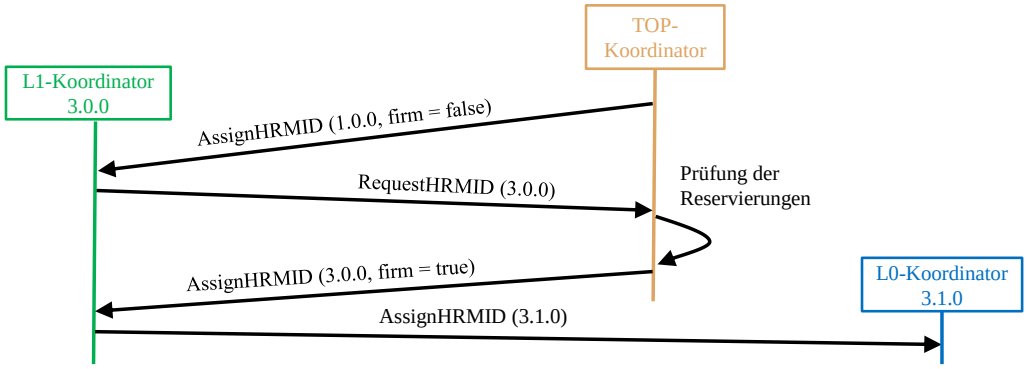

Abbildung 3.28: Signalisierung zur Stabilisierung von Adresszuordnung und Routing

Abbildung 3.28 zeigt die Signalisierung für die Koordinatoren 3.0.0 und 3.1.0 aus Abbildung 3.27. Der *TOP-Koordinator* wurde auf einem anderen Knoten als zuvor platziert und versucht dem untergeordneten L1-Koordinator eine neue Adresse 1.0.0 über eine *AssignHRMID*-Nachricht zuzuweisen. Dieser vergleicht sie mit der lokal gespeicherten Adresse der vorhergehenden Zuweisung und erkennt, dass ein Wechsel stattfinden soll. Er sendet als Reaktion dem übergeordneten Koordinator eine R*equestHRMID*-Nachricht zu, mit der er seine alte Adresse 3.0.0 einfordert. Der empfangende *TOP-Koordinator* verwaltet eine lokale Liste über Adressreservierungen und entscheidet, ob die Adresse bereits durch einen anderen untergeordneten Koordinator reserviert wurde und reagiert entsprechend:

- **HRMID ist nicht reserviert:** Dieser Fall ist in Abbildung 3.28 zu sehen. Die geforderte HRMID ist noch nicht reservier, sodass der *TOP-Koordinator* erneut eine *AssignHRMID*-Nachricht sendet und dabei dieses Mal die zuvor zugewiesene Adresse 3.0.0 signalisiert. Die Nachricht enthält zusätzlich die Markierung „firm“, welche dem Empfänger mitteilt, dass die mitgeteilte Adresse nicht weiter verhandelt werden darf. Der untergeordnete Koordinator behält seine ursprüngliche HRMID und setzt die Adressvergabe innerhalb seines Clusters fort, sodass er beispielsweise dem in Abbildung 3.28 blau dargestellten Koordinator 3.1.0 seine Adresse zuweist.
- **HRMID ist bereits reserviert:** Wurde die HRMID bereits fest zugewiesen, kann sie nicht erneut vergeben werden. Folglich wird eine *AssignHRMID*-Nachricht mit einer neuen HRMID 1.0.0 und der Markierung „firm“ gesendet, sodass eine Adressumstrukturierung tatsächlich durchgeführt wird. Eine erneute Anforderung durch den untergeordneten Koordinator ist nicht erlaubt.

Der vorgestellte Ansatz zur Stabilisierung der Adressen und Routen unterliegt Grenzen. Wenn mehrere Knoten des Netzes plötzlich ausfallen und zu unterschiedlichen Zeiten neustarten, kann dies nur begrenzt kompensiert werden.

3.4.5 Kompatibilität zum Adressierungsschema von IP

Eine HRMID besteht im Allgemeinen aus einer Liste von Nummern, deren Anzahl identisch mit der Tiefe der eingesetzten Hierarchie ist. Bei HRM ist der Wertebereich einer solchen Nummer nicht grundsätzlich begrenzt. Werden aber zusätzliche Beschränkungen im Wertebereich der einzelnen Nummern eingeführt, eröffnen sich neue Möglichkeiten der Kompatibilität zum Internet Protokoll. Die nachfolgenden beiden Abschnitte erläutern, wie HRMIDs in IP-Adressen integriert werden können. Dies bildet die Grundlage für die Erklärungen aus Abschnitt 3.9 zur direkten Interoperation zwischen HRM und IP.

3.4.5.1 Einordnung in IPv4

Entsprechend Abschnitt 2.1.5.1 haben IPv4 Adressen die Form „A.B.C.D“. Neben weiteren Varianten ergeben sich intuitiv 4 Stellen für eine HRMID mit einem Wertebereich von 0 bis 255 pro Stelle. Abschnitt 2.1.5.2 ist zu entnehmen, dass einem Netzwerkoperator der Bereich 10.0.0.0 mit einer Netzmaske

von 255.0.0.0 als größter privater Adressraum zur Verfügung steht. Hierbei bietet sich eine Beschränkung der Hierarchietiefe auf 3 an, sodass für jedes Hierarchielevel 8 Bits der IP-Adresse verwendet werden können. Dadurch wird der Wertebereich eines Hierarchielevels auf 0-255 beschränkt. Zusätzlich muss die Spezifikation für IP-Adressen beachtet werden. HRMIDs dürfen weder der Netzwerk- noch der Broadcast-Adresse des zugewiesenen IP-Adressbereiches entsprechen. Werden diese Einschränkungen beachtet, können HRMIDs sehr einfach nach dem Schema 10.X.Y.Z in IPv4-Adressen umgewandelt werden. Die Werte X, Y und Z stehen dabei stellvertretend für je ein Hierarchielevel. Aus der IP-Adresse 10.3.2.4 ergibt sich folglich die HRMID 3.2.4. Für den umgekehrten Fall muss jeder Knoten den zugewiesenen IP-Adressbereich kennen. Diese Information kann entweder durch eine explizite Voreinstellung zur Startzeit eines Knotens oder durch eine zusätzliche Signalisierung im Netzwerk verteilt werden. Eine Implementierung kann dafür beispielsweise einmalig das Netzwerk mit einer entsprechenden Signalisierungsnachricht fluten.

3.4.5.2 Einordnung in IPv6

Wie im Abschnitt 2.1.5.1 erläutert, folgt der allgemeine Aufbau einer IPv6-Adresse dem Schema „A:B:C:D:E:F:G:H“. Jeder der 8 Buchstaben steht für eine hexadezimale Nummer aus dem Wertebereich 0 bis 0xFFFF. Dieses Schema ermöglicht bis zu 65536 unterschiedliche Werte pro Stelle. Analog zu IPv4 kann auch für IPv6 eine Integration von HRMIDs sehr einfach realisiert werden, indem der Netzwerkoperator dem HRM-Netzwerk ein festes IPv6-Netzwerk zuordnet und diese Konfiguration im Netzwerk verteilt wird. Das kann nach Abschnitt 2.1.5.2 beispielsweise das Netzwerk „fd00:0:0:0:0:0:0:0“ mit einer Präfixlänge von 7 Bits sein. Im Vergleich zu IPv4 ist der schnittstellenspezifische Adressbereich in diesem Fall größer, da er 64 Bits umfasst. Die maximale Tiefe der Kontrollebene kann dadurch im Vergleich zu IPv4 auf 4 Hierarchielevels mit jeweils 16 Bits pro Hierarchielevel erweitert werden, was eine Adressierung von maximal 65536^4 Knoten ermöglicht. Die Umwandlung zwischen beiden Adressierungsschemata erfolgt wiederum ähnlich zu IPv4.

3.4.6 Vergleich der Adressierungsschemata

Die nachfolgende Tabelle 3.4 gibt einen Überblick über die verwendeten Schemata zur Adressierung von Knoten, Kontrollentitäten und Netzwerkschnittstellen für HRM.

	Knoten-ID	**Entität-ID**	**HRMID**
Vergabeziel	Knoten	Entitäten	Netzwerkschnittstellen und Koordinatoren
Verwendung	Festlegung des Quell- oder Zielknotens einer Signalisierungsnachricht der Kontrollebene	Festlegung der Quell- oder Zielentität einer Signalisierungsnachricht der Kontrollebene	Beschreibung von Routen, Angabe der Zieladresse von Paketen
Einsatzgebiet	Kontrollebene	Kontrollebene	Kontrollebene, Datenebene
Eindeutigkeit	global	lokal für Knoten	global
Struktur	flach (automatisch generierte Nummer)	flach (fortlaufende Nummerierung)	hierarchisch (je Hierarchielevel eine Nummer)
Vergabe-authorität	Knoten	Knoten	Koordinatoren der Kontrollebene
Vergabe-zeitpunkt	Start des Knotens	Instanziierung der jeweiligen Entität	sobald übergeordneter Koordinator eine HRMID hat und dessen initiale Pause vorüber ist

Tabelle 3.4: Vergleich zwischen den in HRM verwendeten Adressierungsschemata

Grundsätzlich wird bei HRM zwischen den Adressierungsmechanismen der Kontrollebene und den HRMIDs unterschieden. Erstere sind notwendig, um die allgemeine Signalisierung zwischen Entitäten zu ermöglichen. HRMIDs werden hingegen durch die Kontrollebene zur Beschreibung von Routen verwendet, sodass ein Routing von Anwendungsdaten durch die Datenebene möglich ist.

3.5 Protokoll zur Verteilung von Routingdaten

Für ein dezentrales Routing ist es notwendig, dass Knoten ihr Wissen über ihre lokale Nachbarschaft anderen Knoten in Form von sogenannten Routingdaten mitteilen. Dies muss mit Fokus auf Skalierbarkeit geschehen, sodass das dabei verursachte Signalisierungsaufkommen auch bei größeren Netzwerken möglichst niedrig gehalten wird. Zu diesem Zweck beinhaltet HRM zwei grundsätzliche Ansätze:

- **kurze Kommunikationswege:** Für die Verteilung von notwendigen Topologiebeschreibungen (Linkstatus sowie QoS-Eigenschaften für jeden bekanntem Link) sollten möglichst Kommunikationsbeziehungen mit kurzen Übertragungswegen verwendet werden. Dafür ist die mehrstufige Hierarchie der Kontrollebene vorteilhaft, da bei der Platzierung ihrer Kontrollinstanzen die physikalischen Hopdistanzen zwischen typischen Kommunikationspartnern einbezogen werden und somit die Kommunikationswege möglichst klein gehalten werden. Folglich erfolgt die Verteilung von Routingdaten mit Hilfe der Koordinatorinstanzen im Netzwerk.
- **Aggregation:** Eine geeignete Aggregation der Daten führt zu einer zusätzlichen Reduktion des verursachten Signalisierungsaufkommens. Dies wird für die Kommunikation zwischen den Koordinatoren verwendet und im weiteren Verlauf dieses Abschnittes detailliert erläutert.

Zwischen den Koordinatoren kommt ein *Link-State*-Protokoll zum Einsatz. Im Gegensatz zu *Distance-Vector*- oder *Path-Vector*-Protokollen bietet diese Vorgehensweise den Vorteil einer schnelleren Konvergenz bei Topologieänderungen. Der zentrale Vorteil der hierarchischen Struktur der Kontrollebene kommt dabei zur Anwendung: ein skalierbarer Austausch von *Link-State*-Signalisierungen in Abhängigkeit von dem jeweiligen Hierarchielevel. Die dabei angewandte Aggregation von Topologiedaten erfolgt ebenfalls auf Basis der vorgegebenen Hierarchie. Jeder Netzwerkknoten erhält somit aggregierte Informationen über entfernte Netzwerkabschnitte und speichert diese für nachfolgende Routingentscheidungen innerhalb der Datenebene in seiner lokalen Routingtabelle ab.

Die notwendigen Signalisierungen können entsprechend den Ausführungen von Abschnitt 2.1.6.3 sowohl proaktiv als auch reaktiv ausgeführt werden. Aufgrund der erläuterten Vorteile wird für HRM das proaktive Vorgehen favorisiert. Dadurch werden zum Preis von kontinuierlichen Signalisierungsnachrichten die bei reaktivem Routing typischen Verzögerungen verhindert, sodass auf jedem Knoten zu jeder Zeit möglichst aktuelle Routingtabellen für die Ermittlung einer Routingentscheidung vorliegen.

Der zugrundeliegende Verteilungsprozess besteht grundsätzlich aus drei Phasen:

- **Phase 0:** Sie stellt die Initialisierungsphase dar und beinhaltet ausschließlich Signalisierungen zwischen direkt benachbarten Knoten. Dabei werden Nachrichten ausgetauscht, sodass jeder Knoten Kenntnis über die HRMIDs seiner direkten Nachbarn sowie die QoS-Eigenschaften der Links zu ihnen erhält.
- **Phase 1:** Innerhalb dieser Phase werden Routingdaten aufwärts der Hierarchie mitgeteilt. Jede Entität der Kontrollebene signalisiert ihrem übergeordneten Koordinator aggregierte Daten über die vorhandene physikalische Topologie. Der Prozess endet am TOP-Koordinator. Die empfangenen Routen speichert jeder Koordinator für die nachfolgenden Signalisierungen von Phase 2.
- **Phase 2:** Die letzte Phase besteht ausschließlich aus der Verteilung von Routingdaten abwärts der Hierarchie. Zu diesem Zweck kombiniert jeder Koordinator seine Daten aus Phase 1 mit den knotenlokalen Daten aus Phase 0 zu neuen aggregierten Routen, welche an die jeweiligen

untergeordneten Clustermitglieder signalisiert werden. Der Prozess endet an den Blättern der Hierarchie, sodass letztlich jeder Knoten zusätzliche Routen zu entfernten Netzwerkabschnitten mitgeteilt bekommt und diese in seiner lokalen Routingtabelle abspeichert.

Die folgenden Abschnitte beschreiben die drei Phasen sowie die darin verwendeten Mechanismen zur Aggregation von Routingdaten detailliert. Im Anschluss wird die automatische Aktualisierung von bereits verteilten Routingdaten erläutert.

3.5.1 Phase 0: Bestimmung der lokalen Topologie

Für ein QoS-Routing ist es zwingend erforderlich, die aktuellen Eigenschaften von Links einzubeziehen. Entsprechend Abschnitt 2.2.3.4 verwendet die Kontrollebene für die Beschreibung der QoS-Eigenschaften zum einen die maximal verfügbare Datenrate und zum anderen die minimal zu erwartende Verzögerung[18].

3.5.1.1 Ermittlung der QoS-Eigenschaften von Links

Durch verschiedene Möglichkeiten können die QoS-Eigenschaften einzelner Links bestimmt werden. Der letztlich favorisierte Weg im Kontext realer Szenarien ist implementierungsspezifisch und nicht Bestandteil dieser Arbeit. Die aktuell zu erwartenden Verzögerungen können beispielsweise durch folgende Vorgehensweisen bestimmt werden:

- **Softwaretreiber:** Sofern das lokale Betriebssystem (bei PC-Systemen) oder die Gerätesoftware (bei Routern) diese Information bereitstellen, kann die Verzögerung auch ohne zusätzliche Maßnahmen aus der vorgegebenen Informationen des Herstellers lokal ausgelesen werden.
- **Heuristik:** Auf Basis von ermittelten Daten über die eingesetzte Hardware kann eine Logik zum Einsatz kommen, die einen Wert für die Verzögerung eines Links ableitet. Insbesondere der Verbindungstyp spielt hierbei eine entscheidende Rolle. Für eine kabelgebundene Anbindung sind typischerweise Verzögerungen von 1 ms realistisch. Hingegen kann es bei kabellosen Anbindungen zu Verzögerungen von 50 ms in Abhängigkeit von der eingesetzten Hardware kommen. Als Eingabe der Heuristik kann dabei verwendet werden:
  - **Hardware der lokalen Schnittstelle:** Die Unterscheidung zwischen beiden Verbindungstypen kann über die lokale Systemsoftware und ihre bereitgestellte API erfolgen.
  - **Hardware der Gegenstelle:** Jeder Netzwerkkarte wird vom Hersteller eine MAC-Adresse für den Einsatz von Ethernet auf Schicht 2 zugeordnet. Die darin enthaltene Herstellerkennung zusammen mit der verwendeten gerätespezifischen ID lassen Rückschlüsse auf den Typ der eingesetzten Hardware zu. Dies kann beispielsweise auf die MAC-Adresse der Gegenstelle angewandt werden.
- **Periodische Messung:** Durch den periodischen Versand von Hallo-Paketen, welche die Gegenseite beantwortet, können korrekte Werte für die Verzögerung bestimmt werden. Dazu können beispielsweise die *AnnounceNeighborNode*-Nachrichten aus Phase 0 verwendet werden. In diesem Fall wird die sogenannte *Round Trip Time* (RTT) als Zeit zwischen dem Versand der Anfragenachricht und dem Eintreffen der Antwortnachricht gemessen. Die Hälfte des Wertes entspricht einer sehr guten Näherung der aktuellen Verzögerung entlang des jeweiligen Links.
- **Manuelle Konfiguration:** Alternativ kann auch ein Administrator die Linkeigenschaften definieren.

Die aktuell noch verfügbare Datenrate entlang eines Links kann ähnlich wie die Verzögerung bestimmt werden. Dies ist durch Abfrage von Herstellerinformationen, der Verwendung einer Heuristik oder letztlich auch durch manuelle Konfiguration möglich.

[18] Weitere Parameter wie bspw. Jitter sind für Erweiterungen von HRM denkbar.

3.5.1.2 Ermittlung der HRMIDs von benachbarten Knoten

Zusätzlich zu den QoS-Eigenschaften von vorhandenen Links müssen die erreichbaren HRMIDs des jeweiligen Nachbarknotens ermittelt werden. Diese Beschreibung der lokalen Nachbarschaft eines Knotens kann in den nachfolgenden Phasen zur Bestimmung von Routen zu entfernten Knoten genutzt werden. Zum Austausch der HRMIDs werden Nachrichten des Typs *AnnounceHRMIDs* innerhalb der L0-Cluster verwendet. Sie beinhaltet eine Liste aller lokal vorhandenen HRMIDs. Diese wird an alle umliegenden Nachbarknoten gesendet, sobald sich die HRMID einer lokalen Netzwerkschnittstelle ändert oder ein neuer Nachbarknoten hinzukommt.

3.5.1.3 Erstellung einer Routingtabelle

Nachdem die HRMIDs der Nachbarknoten und die QoS-Eigenschaften der Links zu ihnen bekannt sind, kann die lokale Routingtabelle mit den daraus abgeleiteten Routen befüllt werden. Dies wird nachfolgend an einem Beispiel dargestellt.

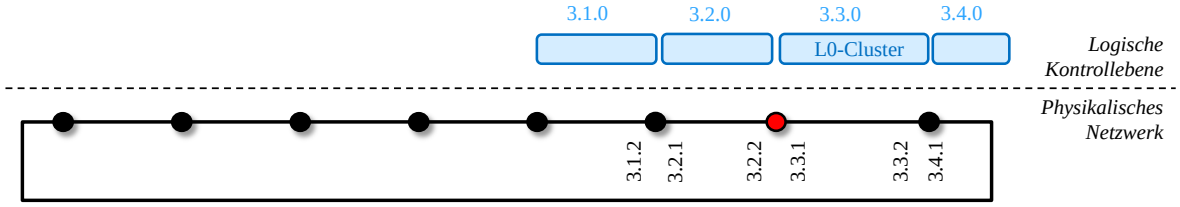

Abbildung 3.29: HRMIDs lokaler Nachbarknoten und -cluster

Abbildung 3.29 stellt das Beispielszenario dar und betrachtet die durch Phase 0 auf dem rot markierten Knoten ermittelten Routen zu den direkten Nachbarn. Dazu zählen die L0-Cluster 3.1.0 und 3.4.0. Beide sind in der Abbildung innerhalb der logischen Kontrollebene durch blaue Bereiche oberhalb der jeweils zugehörigen physikalischen Knoten dargestellt. Ihre Existenz wurde dem rot markierten Knoten von seinen Nachbarknoten durch Austausch von *AnnounceHRMIDs*-Nachrichten mitgeteilt. Zusätzlich erfährt der rot markierte Knoten über diese Nachrichten von der Existenz der HRMIDs 3.2.1 und 3.3.2 seiner direkten Nachbarknoten. Beide gehören zu einem L0-Cluster, für den auch der rot markierte Knoten eine lokale HRMID zugewiesen bekommen hat. Folglich werden sie als clusterinterne Ziele behandelt und nicht über die HRMID ihres übergeordneten L0-Clusters aggregiert.

Ziel	Nächster Router	Hop-Distanz	Datenrate [kbit/s]	Verzögerung [ms]
3.2.1	3.2.1	1	100.000	1
3.3.2	3.3.2	1	100.000	1
3.1.0	3.2.1	1	100.000	1
3.4.0	3.3.2	1	100.000	1

Tabelle 3.5: Erstellte Routingtabelle der lokalen Nachbarschaft

Tabelle 3.5 zeigt die aufgrund der erkannten lokalen Nachbarn erstellten vier Einträge in der lokalen Routingtabelle des ausgewählten Knotens. Sie enthalten sowohl die lokalen Routen zu den zwei clusterinternen HRMIDs der Nachbarknoten als auch jene zu den beiden Nachbarclustern 3.1.0 und 3.4.0. Die aktuell verfügbare Datenrate und die zu erwartende Verzögerung eines Links zu einem Nachbarknoten werden dabei mit 100.000kbit/s bzw. 1 ms beschrieben.

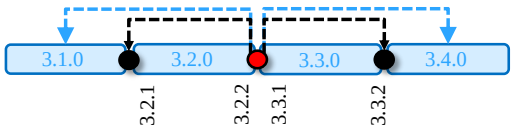

Abbildung 3.30: Bekannte Routen zu Nachbarn auf dem ausgewählten Knoten

Abbildung 3.30 stellt die in Tabelle 3.5 beschrieben Routen nochmals dar, wobei jeder Pfeil an dem jeweiligen Startknoten der Route beginnt und an dem jeweiligen Zielknoten bzw. -cluster endet. Die beiden schwarz dargestellten führen zu Nachbarknoten, welche jeweils einem gemeinsamen L0-Cluster angehören. Des Weiteren sind die blau gestrichelten zu sehen, welche zu den beiden entfernten L0-Clustern 3.1.0 und 3.4.0 führen. Diese Routingdaten stellen die Basis für die nachfolgenden Phasen 1 und 2 dar, welche daraus Routen zu entfernteren Zielen ermitteln.

3.5.1.4 Ergebnis von Phase 0

Nach erfolgter Phase 0 besitzt jeder Knoten Kenntnis über seine lokale Nachbarschaft. Dazu zählen sowohl die HRMIDs aller Nachbarknoten als auch die QoS-Eigenschaften der Links zu den jeweiligen Nachbarn. Diese Daten der lokalen Nachbarschaft sind in der lokalen Routingtabelle jedes Knotens in Form von Routen gespeichert und bilden die Basis für die nachfolgenden beiden Phasen.

3.5.2 Phase 1: Report von lokalen Routen

In Phase 1 signalisiert jeder Knoten seine lokalen Topologiedaten an seinen übergeordneten Koordinator, welcher seine Daten wiederum dem ihm übergeordneten mitteilt. Dieser Prozess wird solange wiederholt, bis der *TOP-Koordinator* die Topologiebeschreibungen empfängt. Zur Signalisierung werden dabei sogenannte *RouteReport*-Nachrichten mit folgendem Inhalt verwendet:

1.) **Routen zu den direkten Nachbarn:** Dabei kann es sich entweder um einen direkten Nachbarknoten oder -cluster handeln. Im letzteren Fall wird eine Zielaggregation für alle Knoten des Nachbarclusters verwendet.
2.) **Routen zur Clusterquerung:** Dies wird für das Routing zur Durchquerung des jeweils untergeordneten Clusters benötigt. Ein Koordinator signalisiert dabei Routen, welche jeweils von einem Gateway zu einem anderen führen. Eine detailliertere Beschreibung dieser Aggregation wird im Abschnitt 3.5.2.2 gegeben.

Aufgrund der verwendeten Zielaggregation für clusterinterne Ziele enthalten *RouteReport*-Nachrichten ausschließlich die oben aufgeführten Typen von Einträgen. In den nachfolgenden beiden Abschnitten werden jeweils Details zu beiden möglichen Typen gegeben.

3.5.2.1 Report von Routen zu den direkten Nachbarknoten und –clustern

Nachfolgend wird der Report anhand eines ausgewählten Beispiels erläutert. Dabei wird zuerst ein Überblick über die Kommunikationsschritte gegeben, anschließend werden die Inhalte der Nachrichten für ausgewählte Teile der Hierarchie erläutert und die Beschreibung von Routen am Beispiel von ausgewählten *RouteReport*-Nachrichten gezeigt.

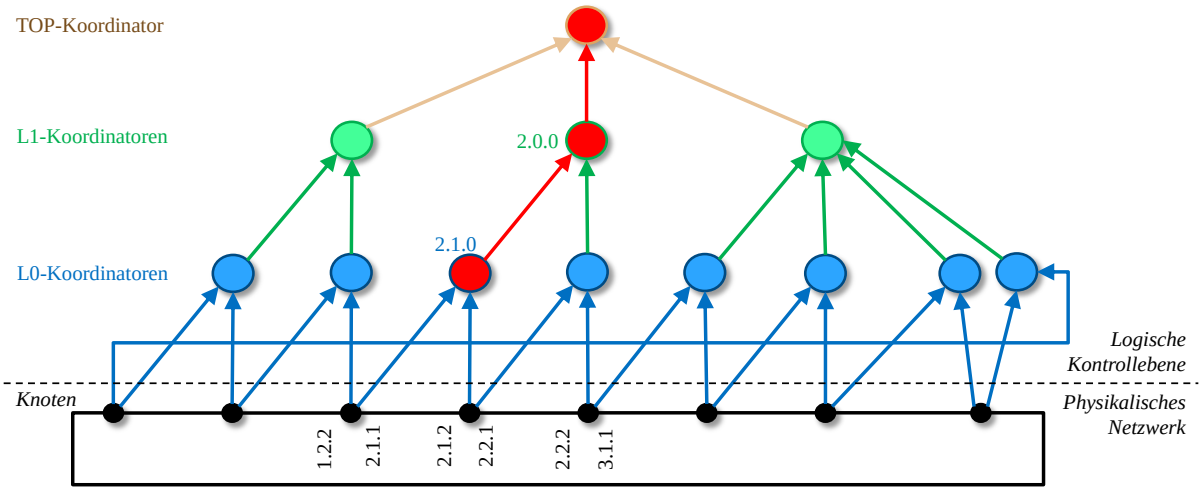

Abbildung 3.31: Signalisierung von *RouteReport* Nachrichten an übergeordnete Koordinatoren

Abbildung 3.31 zeigt die allgemeine Signalisierung von *RouteReport*-Nachrichten für das Beispielszenario. Die Knoten des L0-Clusters 2.1.0 besitzen dabei aufgrund von Phase 0 bereits Kenntnis über ihre direkte Nachbarschaft und haben die zugehörigen Routingdaten an ihren übergeordneten Koordinator 2.1.0 signalisiert. Dieser sendet die empfangenen Topologiedaten in aggregierter Form an seinen übergeordneten Koordinator 2.0.0, welcher wiederum seine von beiden untergeordneten Koordinatoren empfangenen Routingdaten in aggregierter Form an den TOP-Koordinator verschickt. Beide Signalisierungsschritte sind anhand der rot markierten Pfeile in Abbildung 3.31 zu erkennen. Der TOP-Koordinator erhält somit Kenntnis über die direkten Nachbarn des Clusters 2.0.0 und kann diese Information für Phase 2 verwenden.

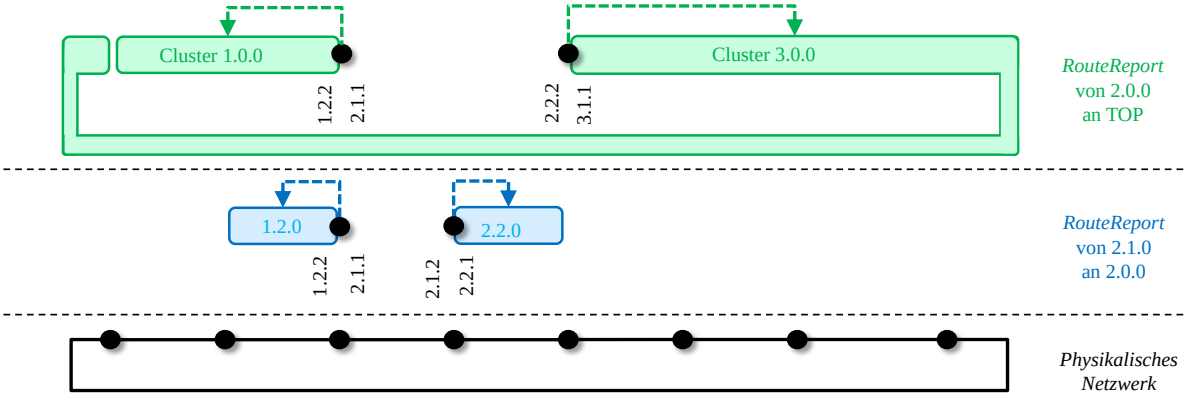

Abbildung 3.32: Routen zur direkten Nachbarschaft

Abbildung 3.32 stellt im unteren Teil die physikalische Topologie dar. Darüber sind für die Hierarchielevel 0 und 1 die ausgewählten Topologiedetails zu sehen, welche in den *RouteReport*-Nachrichten an den jeweils übergeordneten Koordinator signalisiert werden. Alle 3 Ebenen sind dabei exakt so übereinander angeordnet, dass sich die physikalischen Knoten exakt unterhalb ihrer Entsprechungen aus den Nachrichten befinden. Jeder Pfeil in den oberen beiden Eben steht stellvertretend für je eine mitgeteilte Route. Ausgangspunkt ist dabei immer der Startknoten und Endpunkt ist der Zielcluster des jeweils beschriebenen Wegs – es wird also stets eine Zielaggregation angewandt. Zu den signalisierten Routingdaten gehören für den dargestellten Fall:

- **Hierarchielevel 0:** Der Koordinator 2.1.0 teilt seinem übergeordneten Koordinator 2.0.0 die Routen zu den angrenzenden Nachbarclustern 1.2.0 und 2.2.0 mit. Erstere startet an dem Knoten mit der clusterintern HRMID 2.1.1 und letztere startet bei dem Knoten mit der clusterinternen

Adresse 2.1.2. Beide Knoten stellen somit Gateways des Clusters 2.1.0 zu je einem Nachbarcluster dar.

- **Hierarchielevel 1:** Von Koordinator 2.0.0 wird an den TOP-Koordinator jeweils eine Route zu den Nachbarclustern 1.0.0 und 3.0.0 gemeldet. In beiden Fällen liegt der Startpunkt auf dem jeweiligen Gateway-Knoten, welcher mindestens eine zugewiesene clusterinterne HRMID hat.

Signalisierungsrichtung	Lokale Route			Hop-Distanz	Datenrate	Verzögerung
	Start	Ziel	Nächster Router			
2.1.0 ➔ 2.0.0	2.1.1	1.2.0	1.2.2	0	unbegrenzt	0
	2.1.2	2.2.0	2.2.1	0	unbegrenzt	0
2.0.0 ➔ TOP	2.1.1	1.0.0	1.2.2	0	unbegrenzt	0
	2.2.2	3.0.0	3.1.1	0	unbegrenzt	0

Tabelle 3.6: Routen zu Nachbarn in *RouteReport*-Nachrichten für verschiedene Hierarchielevels

In Tabelle 3.6 ist die in den jeweiligen *RouteReport*-Nachrichten verwendete Repräsentation der in Abbildung 3.32 aufgeführten Routen zu sehen. Dabei ist zu erkennen:

- Eine Route zu einem direkten Nachbarcluster wird anhand des Übergangs zwischen beiden angrenzenden Clustern beschrieben. Beispielsweise stellt die erste Zeile die Route zum Nachbarcluster 1.2.0 dar, welche auf Basis des knotenlokalen Links zwischen 2.1.1 und 1.2.2 beschrieben wird. Mit der Adresse 1.2.2 wird der nächste Router in Richtung des Zielclusters 1.2.0 identifiziert.
- In Abhängigkeit vom jeweiligen Hierarchielevel wird eine Zielaggregation für die HRMID jedes Nachbarclusters verwendet und zur Beschreibung der über diese Route erreichbaren Ziele verwendet. Diese Form der Aggregation ist möglich, da aufgrund des Algorithmus zur Clusterbildung aus Abschnitt 3.3.4 ausschließlich zusammenhängende Cluster erstellt und aufrechterhalten werden.
- Die zugeordneten Eigenschaften der Route enthalten eine Hop-Distanz mit dem Wert 0, da ausschließlich knotenlokale Links zum Erreichen des angegebenen Ziels verwendet werden. Die Verzögerung auf einem Knoten wird aus diesem Grund ebenfalls mit 0 festgelegt. Dagegen wird die maximal mögliche Datenrate über die beschriebene Route als unbegrenzt festgelegt.

3.5.2.2 Report von Routen zur Clusterquerung

Würden stets alle internen Routen eines Clusters in *RouteReport*-Nachrichten kommuniziert werden, wäre ein hoher Signalisierungsaufwand die Folge. Zur Vermeidung wird bei HRM eine Abstraktion von physikalischen Netzwerkstrukturen verwendet. Der damit verbundene Prozess wird entsprechend Abschnitt 2.2.5 als Topologieaggregation bezeichnet. Eine daraus resultierende aggregierte Route kann aus verschiedenen physikalischen Links bestehen und sich über verschiedene Knoten erstrecken. Für HRM wird eine Maschentopologie[19] verwendet, wodurch aggregierte Routen ausschließlich zwischen Paaren von Gateway-Knoten eines Clusters ermittelt werden. Da in dieser Arbeit die einfache (statt der multidimensionalen) Aggregation favorisiert wird, muss eine Priorisierung zwischen den Eigenschaften der Routen eingeführt werden. Für HRM wird dabei als primäres Attribut die aktuell verfügbare Datenrate und als sekundäres die aktuell zu erwartende Verzögerung entlang der physikalischen Pfade verwendet und daraus die letztlich aggregierte Route zwischen zwei Gateways bestimmt. Diese Vorgehensweise stellt aus des Autors dieser Arbeit einen akzeptablen Kompromiss zwischen Datenreduktion und verbleibender Genauigkeit der Routingdaten dar.

[19] Siehe Variante A3 in Abschnitt 2.2.5.

3.5.2.3 Ergebnis von Phase 1

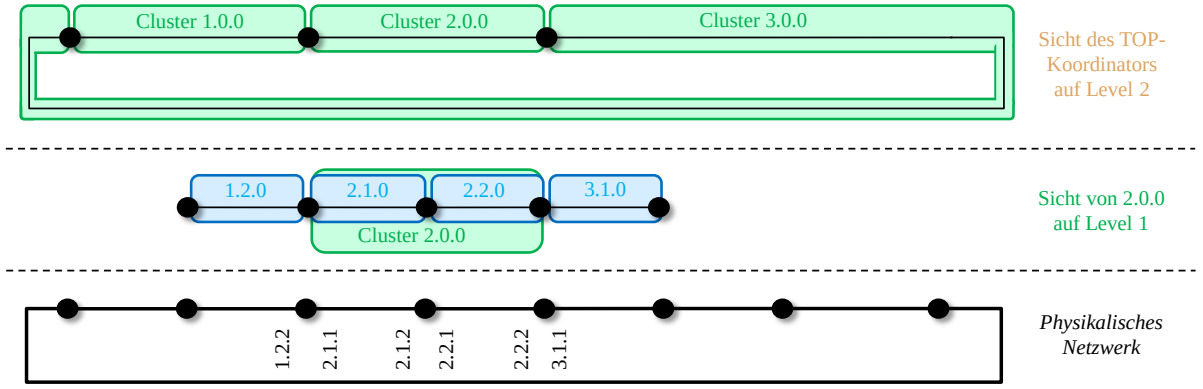

Abbildung 3.33: Topologieansicht auf verschiedenen Hierarchielevels nach Empfang von *RouteReport*-Nachrichten

Abbildung 3.33 zeigt die durch *RouteReport*-Nachrichten resultierenden Routingdaten für den TOP-Koordinator sowie den untergeordneten Koordinator 2.0.0 im Vergleich zur physikalischen Netzwerktopologie:

- **TOP-Koordinator:** Nach Empfang aller Nachrichten von Phase 1 besitzt der *TOP-Koordinator* eine aggregierte Beschreibung der gesamten Netzwerktopologie. Dazu zählen alle direkten Verbindungen zwischen seinen untergeordneten Clustern sowie deren aggregierte interne Topologie. Für das Beispielnetzwerk ist dies jeweils nur eine aggregierte Route zwischen den Gateways.
- **Koordinator 2.0.0:** Dieser Koordinator verfügt nach Ablauf von Phase 1 nur begrenzte Routingdaten und kennt seine direkten Nachbarcluster 1.2.0 sowie 3.1.0. Des Weiteren kennt er die knotenlokalen Routen zwischen seinen untergeordneten Clustern 2.1.0 und 2.2.0. Zusätzlich kennt er deren aggregierte interne Topologie, welche ihm durch seine untergeordneten Koordinatoren signalisiert wurde. Für das dargestellte Netzwerk besteht dies aus je einer Route zwischen den Gateways.

Allgemein betrachtet werden durch Phase 1 die existierenden Routen, inklusive ihrer QoS-Eigenschaften, in Richtung des *TOP-Koordinators* mit Hilfe der Hierarchie signalisiert, wobei mit zunehmendem Hierarchielevel die Routingdaten immer weiter aggregiert werden. Als Resultat hat jeder Koordinator Routingdaten über seine untergeordneten Cluster und kennt vorhandene Routen zu direkten Nachbarn.

3.5.3 Phase 2: Verteilung von Routen zu entfernten Zielen

Nach Phase 1 hat ein Koordinator ohne weitere Signalisierung keine Kenntnis über alle Routen zu entfernten Knoten, welche nicht zu direkten Nachbarclustern gehören. Phase 2 hat die Aufgabe, diese Routen durch Kombination bekannter Daten aus Phase 0 und 1 zu ermitteln und im Netzwerk zu signalisieren. Dadurch können die vorhandenen Routingtabellen im Netzwerk erweitert werden, sodass ein Routing für alle Zieladressen lückenlos ermöglicht wird.

Ausgehend vom *TOP-Koordinator* werden in Phase 2 neue Routen in Richtung der Blätter der Hierarchie mitgeteilt. Dabei sendet jeder Koordinator eine sogenannte *RouteShare*-Nachricht an jedes Mitglied seines untergeordneten Clusters. Eine solche Nachricht beinhaltet die folgenden Einträge:

1.) **Routen zu Geschwisterclustern höherer Hierarchielevels:** Außer dem TOP-Koordinator empfängt jeder Koordinator ebenfalls *RouteShare*-Nachrichten seines übergeordneten Koordinators. Diese speichert er separat ab. Für Signalisierungen an sein jeweiliges Clustermitglied kombiniert er diese mit der ihm bekannten clusterinternen Topologie (welche durch vorherige *RouteReport*-Nachrichten mitgeteilt wurde) zu neuen Routen, deren Startpunkt das jeweilige Clustermitglied ist.

2.) **Routen zu clusterinternen Geschwistern**
Der Koordinator bestimmt Routen zu anderen Mitgliedern seines Clusters. Dabei sind zwei verschiedene Varianten möglich:

a. **Routing durch den Cluster:** Es werden Routen bestimmt, welche ausschließlich interne Pfade des Clusters verwenden.
b. **Routingschleifen:** Es werden Routen bestimmt, welche ebenfalls Pfade zu Nachbarclustern verwenden, um clusterinterne Ziele zu erreichen. Dies stellt eine sogenannte *Routingschleife* dar.

Ähnlich zu Phase 1 wird für *RouteShare*-Nachrichten eine Zielaggregation für alle Knoten eines Clusters verwendet. Die resultierende Genauigkeit ist dabei abhängig von dem jeweiligen Hierarchielevel der Aggregation und wird im Folgenden an einem Beispiel verdeutlicht. Die zuvor in Phase 1 eingeführte Priorisierung zwischen Datenrate und Verzögerung kommt dabei ebenfalls zum Einsatz, sodass die Datenrate als primäre QoS-Eigenschaft verwendet wird.

3.5.3.1 Kombination und Verteilung neuer Routen

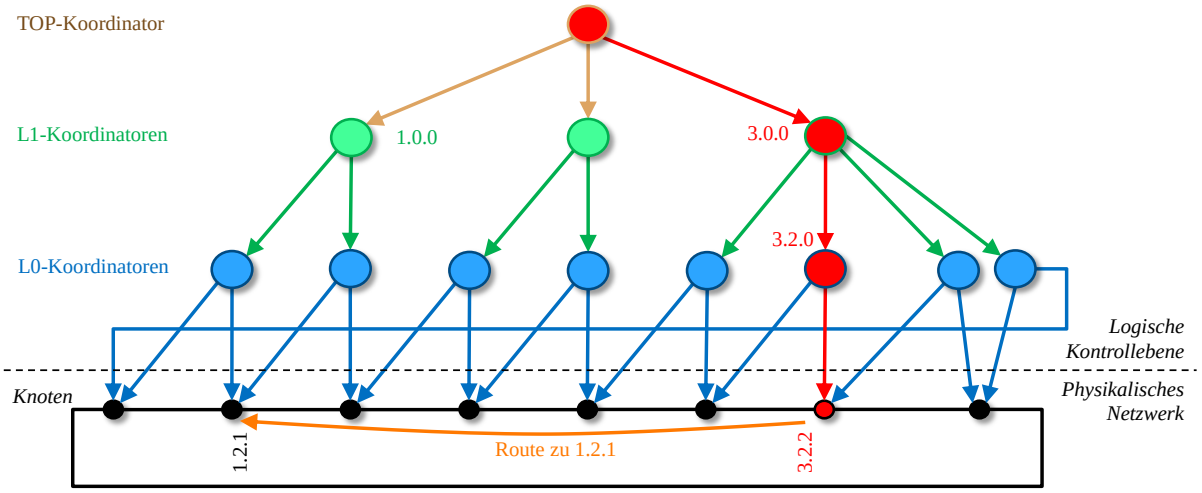

Abbildung 3.34: Signalisierung von *RouteShare*-Nachrichten (Pfeile) von oben nach unten in der Kontrollhierarchie

Abbildung 3.34 zeigt Phase 2 für das Beispielnetzwerk. Der rot markierte Knoten mit der HRMID 3.2.2. erhält auf Basis von *RouteShare*-Nachrichten, ausgehend vom *TOP-Koordinator*, über die Koordinatoren 3.0.0 und 3.2.0 zusätzliche Routen mitgeteilt. Als Resultat dieser Signalisierung kennt der Knoten eine Route, welche Pakete zur Adresse 1.2.1 über einen Pfad durch Cluster 2.0.0 an den Zielcluster 1.0.0 leitet. Um zu klären, welche neuen Informationen über die Netzwerktopologie die signalisierten Routingdaten zwischen den unterschiedlichen Hierarchielevels bieten, werden die aufgrund von Phase 2 zusätzlich bekannten Routen im Folgenden für unterschiedliche Hierarchielevels betrachtet.

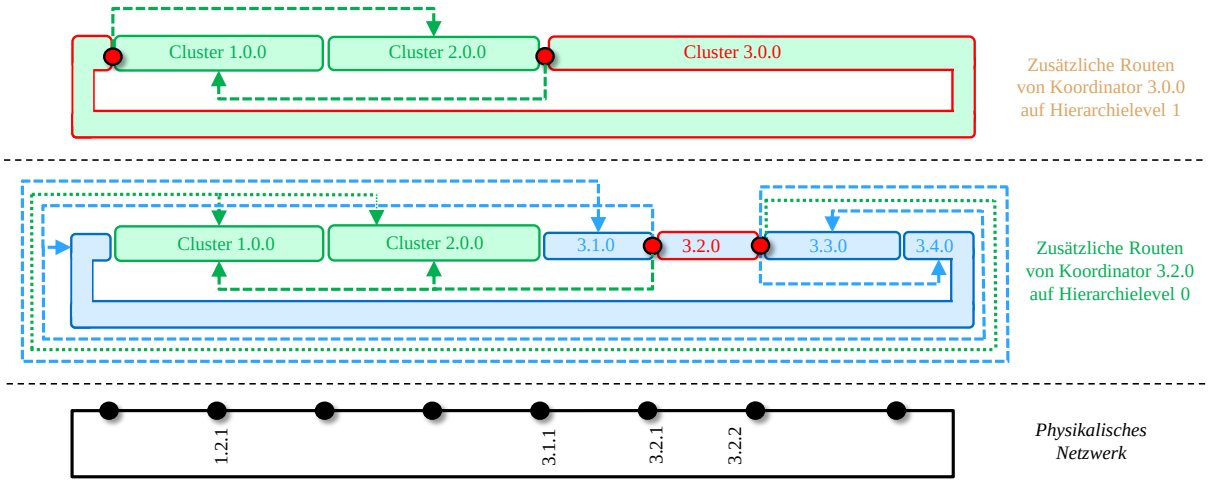

Abbildung 3.35: Zusätzliche Routen für Koordinatoren auf Level 1 und 0 nach Empfang von *RouteShare*-Nachrichten

Abbildung 3.35 zeigt für Hierarchielevel 1 und 0 anhand der Koordinatoren 3.0.0 und 3.2.0 die durch *RouteShare*-Nachrichten zusätzlich bekannten Routen:

- Der Koordinator 3.0.0 erhält Kenntnis über die grün dargestellten Routen (Pfeile) zu seinen Nachbarclustern 1.0.0 und 2.0.0.
- Der darunter liegende Koordinator 3.2.0 erhält vier Routen zu den Clustern 1.0.0 und 2.0.0 seines übergeordneten Hierarchielevels sowie zusätzliche Routen zu seinen Geschwisterclustern. Alle Routen starten hierbei wiederum auf den rot dargestellten Gateway-Knoten des Clusters 3.2.0.

Folgender Ablauf teilt dabei dem Knoten mit der HRMID 3.2.2 eine Route für das Ziel 1.2.1 mit:

1.) **TOP-Koordinator ➔ Koordinator 3.0.0:** Der TOP-Koordinator bestimmt alle clusterintern Geschwister von 3.0.0 und ermittelt jeweils Routen zu ihnen[20]. Dabei wird eine Route von 3.0.0 nach 1.0.0 durch den Cluster 2.0.0 ermittelt, welche auf dem Knoten mit der HRMID 3.1.1 startet.
2.) **Koordinator 3.0.0 ➔ Koordinator 3.2.0:** Der Koordinator 3.0.0 verwendet die zuvor vom TOP-Koordinator empfangene Route (von 3.0.0 via 2.0.0 zu 1.0.0), um eine neue Route für den Koordinator 3.2.0 auf Basis seiner lokalen Daten zu schlussfolgern. Dafür verwendet er sein Wissen über die interne Topologie seines Clusters, welche ihm durch vorhergehende *RouteReport*-Nachrichten bekannt ist. Durch die Kombination beider Datenbasen kann er eine neue Route via Cluster 2.0.0 zu 1.0.0 bestimmen, welche jedoch bei HRMID 3.2.1 startet und somit ebenfalls den Cluster 3.1.0 passiert.
3.) **Koordinator 3.2.0 ➔ Clustermitglied 3.2.2:** Der Koordinator 3.2.0 führt den gleichen Prozess wie Koordinator 3.0.0 aus und ermittelt dadurch eine Route via Cluster 2.0.0 zu 1.0.0, welche jedoch bei HRMID 3.2.2. startet und somit neben dem Cluster 3.1.0 auch den Cluster 3.2.0 passiert.

Als Ergebnis erhält der Knoten mit der HRMID 3.2.2 eine Route zu Zielen in Cluster 1.0.0, welche den Cluster 2.0.0 passiert. Dies stellt jedoch nur ein ausgewähltes Beispiel dar, die Signalisierungen werden allgemein zwischen alle Koordinatoren und für alle bekannten Routen durchgeführt.

[20] In Abschnitt 4.2.6 wird dazu erläutert, wie multiple Routen zwischen zwei HRMIDs innerhalb einer Implementierung gefunden werden können.

3.5.3.2 Ergebnis von Phase 2

Das Ergebnis von Phase 2 wird beispielhaft für den Knoten mit HRMID 3.2.2 genauer betrachtet.

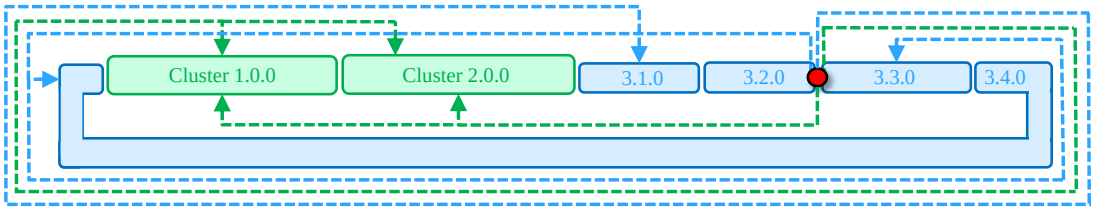

Abbildung 3.36: Resultierende zusätzliche Routen für einen Beispielknoten nach Empfang von *RouteShare*-Nachrichten

Abbildung 3.36 stellt alle Routen dar, welche aufgrund von *RouteShare*-Nachrichten von Koordinator 3.2.0 an den Knoten mit HRMID 3.2.2 signalisiert werden. Dabei ist zu erkennen, dass ihm nach erfolgter Phase 2 beide jeweils existierenden Routen zu den Zielen des Beispielnetzwerks bekannt sind.

Ziel	Nächster Router	Hop-Distanz	Datenrate [kbit/s]	Verzögerung [ms]
3.2.1	3.2.1	1	100.000	1
3.3.2	3.3.2	1	100.000	1
3.1.0	3.2.1	1	100.000	1
3.4.0	3.3.2	1	100.000	1
3.4.0	3.2.1	6	100.000	6
3.3.0	3.2.1	7	100.000	7
3.1.0	3.3.2	6	100.000	6
1.0.0	**3.2.1**	**4**	**100.000**	**4**
2.0.0	3.2.1	2	100.000	2
1.0.0	3.3.2	2	100.000	2
2.0.0	3.3.2	4	100.000	4

Tabelle 3.7: Routingtabelle des Knotens mit der HRMID 3.2.2

Tabelle 3.7 zeigt eine Übersicht über die resultierende lokale Routingtabelle des ausgewählten Knotens[21]. Bereits aus Phase 0 sind die grau hinterlegten Einträge bekannt. Darunter sind, blau und grün hinterlegt, die durch die Kontrollebene signalisierten sieben zusätzlichen Routen aufgeführt. Diese stehen ausschließlich aufgrund von *RouteReport*- und *RouteShare*-Nachrichten dem Knoten zur Verfügung und können somit auch durch den Routingalgorithmus der Datenebene für notwendige Routingentscheidungen berücksichtigt werden. Darin enthalten sind im oberen Teil die drei Routen zu den L0-Clustern 3.4.0, 3.3.0 und 3.1.0. Die unteren vier Einträge beschreiben die jeweils zwei existierenden Routen zu den beiden L1-Clustern 1.0.0 sowie 2.0.0. Dazu zählt insbesondere der rot markierte Tabelleneintrag, welcher Pakete mit dem Ziel 1.2.1 entlang des Clusters 2.0.0 in Richtung des Zielclusters 1.0.0 leitet. Es ist zu erkennen, dass die darin beschriebene Routenlänge von 4 wesentlich größer als jene der Alternativroute über 3.3.2 ist. Zusätzlich sind in den hinteren beiden Spalten die signalisierten QoS-spezifischen Eigenschaften der Routen zu erkennen. Die aktuell verfügbare Datenrate und die zu erwartende Verzögerung aller Links werden dabei mit 100.000kbit/s bzw. 1 ms angenommen, sodass unter diesen

[21] Zur besseren Nachvollziehbarkeit der Tabelle sind die HRMIDs des jeweils nächsten Routers entsprechend der Richtung im Netzwerk ausgerichtet. Die Details zu den QoS-Eigenschaften wurden an dieser Stelle absichtlich ausgeblendet.

Bedingungen die resultierende Gesamtverzögerung gleich der Hop-Distanz für jede übermittelte Route ist.

Allgemein betrachtet berechnet in Phase 2 jeder Koordinator zusätzliche Routen auf Basis der durch die Phasen 0 und 1 erhaltenen Topologiedaten. Diese teilt er allen untergeordneten Clustermitgliedern mit. Als Ergebnis von Phase 2 besitzt jeder physikalische Knoten zusätzliche vorberechnete Routen zu entfernten Zielen im Netzwerk und speichert diese in seiner lokalen Routingtabelle ab.

3.5.4 Aktualisierung von Routingdaten

In einem Netzwerk können spontan Links wegfallen oder hinzukommen. Zusätzlich können sich die Eigenschaften vorhandener Links ändern, sodass beispielsweise die verfügbaren Datenraten einzelner Routen variieren. Diese Änderungsereignisse müssen kontinuierlich in Phase 0 erkannt und unter den Knoten mit Hilfe der Kontrollebene kommuniziert werden. Für ein skalierbares Management ist es dabei notwendig, die verursachte Datenrate der Signalisierungen gering zu halten. Dennoch müssen die Aktualisierungen häufig genug stattfinden, um möglichst aktuelle Routingdaten auf allen Knoten zu gewährleisten.

3.5.4.1 Teil- und Vollaktualisierungen

Innerhalb der Kontrollebene werden zur Reduktion des Datenmehraufwands neben vollständigen Aktualisierungen auch Teilaktualisierungen verwendet. Ähnlich den Signalisierungen von BGP enthalten sie nur jene Routen, deren Eigenschaften sich seit der letzten Aktualisierungsnachricht verändert haben. Diese Form der Datenreduktion wird sowohl für *RouteReport*- als auch *RouteShare*-Nachrichten angewandt. Um stets eine vollständige Repräsentation der Routingdaten zur Verfügung zu haben, muss der Empfänger jeweils lokal eine vollständige Sicht über alle bekannten Routen speichern.

Zur Sicherstellung konsistenter Routingdaten werden zusätzlich Vollaktualisierungen entsprechend eines definierten Intervalls gesendet. Sie stellen den mindestens erforderlichen Datenverkehr zur Synchronisation von Routingdaten im Netzwerk dar.

3.5.4.2 Aktualisierungsintervalle

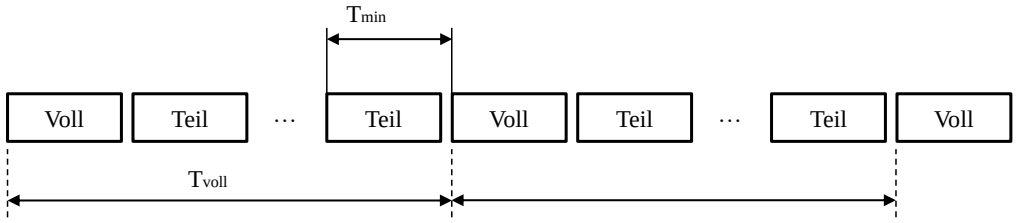

Abbildung 3.37: Zeitintervalle von Teil- und Vollaktualisierungen

Abbildung 3.37 zeigt die Intervalle für Teil- und Vollaktualisierungen von Routingdaten. Der Wert T_{min} legt dabei die Zeit fest, die zwischen zwei aufeinander folgenden Aktualisierungen minimal vergehen muss. Hierdurch werden die maximale Häufigkeit von Signalisierungen und die dadurch verursachte Datenrate begrenzt. Treten Änderungen an der Topologie oder den Linkeigenschaften in kürzeren Abständen auf, dürfen die Nachrichten frühestens im Abstand T_{min} zur letzten Aktualisierung verschickt werden. Als Gegenstück dazu legt der Wert T_{voll} die Zeit zwischen zwei aufeinanderfolgenden Vollaktualisierungen fest. Wurde zuletzt eine solche Vollaktualisierung versandt, muss bis zur nächsten Teilaktualisierung mindestens die Zeit T_{min} vergehen. Der dafür zu wählende Wert bestimmt somit die maximale Verzögerung bis ein Knoten Kenntnis über eine Änderung im Netzwerk erhält. Die Häufigkeit von Vollaktualisierungen sollte wiederum von der Zuverlässigkeit des Netzwerks abhängen. Weitere Details zu dem für HRM verwendeten Intervallen werden im Kontext der empirischen Evaluation in Abschnitt 6.3 gegeben.

3.5.4.3 Alterung von Routingdaten

Sollten Knoten oder Links physikalisch ausfallen, fallen aus Sicht des Routings ebenfalls in den betroffenen Netzteilen mögliche Routen für das restliche Netzwerk aus. Die Erkennung lokaler Linkausfälle wurde in Abschnitt 3.3.8 betrachtet. Routingdaten müssen in diesem Fall aktualisiert werden.

Bei Routenausfall wird bei HRM für eine kurze Konvergenzzeit eine explizite Signalisierung innerhalb der Kontrollebene verwendet, um eine automatische Aktualisierung der Routingdaten im Netzwerk zu veranlassen. Sie wird vom erkennenden Router durch Phase 1 gestartet, sodass er an übergeordnete Koordinatoren aktualisierte Routen übermittelt. Da Phase 2 die Daten von Phase 1 als Basis für die Routenberechnungen verwendet, werden folglich durch Phase 2 ebenfalls aktualisierte Routingdaten verteilt.

Da eine topologische Veränderung an verschiedenen Stellen des Netzwerks gleichzeitig stattfinden kann, ist eine Unterbrechung von Signalisierungskanälen innerhalb der Kontrollebene denkbar. Der explizite Signalisierungsmechanismus genügt somit nicht zur Sicherstellung konsistenter Routingdaten. Zur Lösung wird zusätzlich die Alterung von Routingdaten eingeführt. Dazu wird jedem Datum in *RouteReport*- als auch *RouteShare*-Nachrichten eine Zeitangabe zur Begrenzung der Gültigkeit hinzugefügt.

$$t_{delete} = t_{receive\_time} + t_{life\_time} + t_{delay\_E2E}$$

Formel 3.6: Berechnung der Zeit für die Löschung einer empfangenen Route

Die Gültigkeit einer Route wird mit Hilfe einer relativen Zeitangabe innerhalb der Signalisierungsnachrichten festgelegt, sodass der Empfänger den absoluten Zeitpunkt für die Löschung der Route nach Formel 3.6 lokal berechnet. Dieses Vorgehen vermeidet eine globale Zeitsynchronisation über Protokolle wie das *Network Time Protocol* (NTP) [105]. Formel 3.6 verwendet stattdessen folgende Werte:

- $t_{receive\_time}$**:** Die lokale Zeit des Empfangs der Nachricht wird an dieser Stelle einbezogen.
- t_{life_time}**:** Die Zeitspanne, in welcher die übermittelte Route ihre Gültigkeit behält, wird durch diesen Wert beschrieben.
- t_{delay_E2E}**:** Die auftretenden Übertragungsverzögerungen im Netzwerk werden – ähnlich Abschnitt 3.3.7 – durch diesen konstanten Wert abgebildet. Er gibt die maximal zu erwartete Ende-zu-Ende-Verzögerung während einer Signalisierung an.

Sobald die lokale Uhrzeit einen Wert größer oder gleich t_{delete} annimmt, gilt die jeweilige Route als veraltet. Sie wird lokal gelöscht und fließt zukünftig nicht mehr in die Verteilung von Routingdaten ein. Des Weiteren steht sie nicht mehr für Routingberechnungen zur Verfügung. Um eine Route vor ihrer automatischen Löschung zu bewahren, müssen die Aktualisierungen aus Phase 1 und 2 zyklisch am Empfänger eintreffen, sodass die Zeit t_{delete} ständig aktualisiert wird.

3.6 Nachrichtenformate der Kontrollebene

Innerhalb der Kontrollebene existieren drei eigenständige Prozesse zur Synchronisation der Instanzenplatzierung, Adressvergabe und Verteilung von Routingdaten. Dabei sind sowohl Punkt-zu-Punkt als auch Punkt-zu-Mehrpunkt basierende Signalisierungen im Einsatz. Im Folgenden werden die zugehörigen Nachrichtenformate beider Signalisierungstypen erläutert. Im Anhang 0 ist zusätzlich eine separate Spezifikation jedes Nachrichtentyps der Kontrollebene enthalten.

3.6.1 Punkt-zu-Punkt-Signalisierungen

Für Punkt-zu-Punkt Signalisierungen der Kontrollebene wird ein Transportprotokoll benötigt. Es muss mindestens folgende Funktionen bereitstellen:

- **Adressierung des Zielknotens:** Der Zielknoten einer Signalisierungsnachricht muss eindeutig festgelegt werden können, sodass sie dem gewünschten Knoten zugestellt wird.
- **Adressierung der Entitäten:** Die Signalisierungsnachricht muss auf Empfängerseite der korrekten Entität zugestellt werden. Des Weiteren muss diese die Möglichkeit haben, die Quelle der Signalisierung zuzuordnen. Folglich müssen sowohl die Ziel- als auch die Quellentität eindeutig identifizierbar sein.
- **Transportsicherung**: Beide Kommunikationspartner müssen sich auf die Kommunikation verlassen können. Dies ist eine Annahme der drei Protokolle der Kontrollebene. Das verwendete Transportprotokoll muss eventuell auftretende Nachrichtenfehler erkennen und kompensieren können.
- **Nachrichtenreihenfolge**: Ebenfalls ist es notwendig, dass Nachrichten in der gesendeten Reihenfolge auf Empfängerseite zugestellt werden. Dies ist ebenfalls eine Annahme der Kontrollebene. Ein Koordinator könnte andernfalls Daten eines übergeordneten Clustermanagers ignorieren, da die zugehörige *RequestClusterMembership*-Nachricht zum Start der Kommunikation verspätet eintrifft.

3.6.1.1 Verbindungen und Kanäle

Punkt-zu-Punkt Signalisierungen werden stets zwischen zwei Entitäten ausgetauscht. Beide sind jeweils auf einem Knoten des Netzwerks instanziiert. Daraus ergeben sich 2 Stufen für Adressierung und Routing[22]. Die erste Stufe verwendet die Knoten-ID des Zielknotens, die zweite benötigt die Entität-ID der Zielinstanz auf dem Zielknoten. Diese Zweiteilung wird ebenfalls auf Kommunikationsverbindungen der Kontrollebene angewandt, sodass grundsätzlich zwischen Verbindungen und Kanälen unterschieden wird. Eine Verbindung besteht zwischen zwei Knoten und es darf maximal eine für jedes Knotenpaar existieren. Dagegen darf jede Verbindung beliebig viele Kommunikationskanäle beinhalten. Jeder Kanal ermöglicht dabei Signalisierungen zwischen zwei Entitäten der Kontrollebene. Diese befinden sich jeweils auf den Knoten an den beiden Enden der Verbindung.

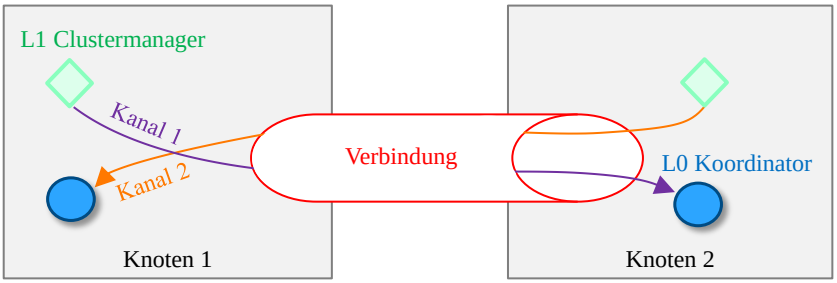

Abbildung 3.38: Eine Verbindung zwischen zwei Knoten mit zwei Kommunikationskanälen zwischen verschiedenen Entitäten der Kontrollebene

Abbildung 3.38 stellt beispielhaft eine solche Kombination aus einer Verbindung und zwei Kanälen dar. Der L1-Clustermanager auf Knoten 1 hat eine Kommunikation mit dem untergeordneten L0-Koordinator auf Knoten 2 gestartet. In Gegenrichtung wurde eine ähnliche Kommunikation durch den L1-Clustermanager auf Knoten 2 zum L0-Koordinator auf Knoten 1 gestartet. Dies führt zu zwei separaten Kanälen, welche beide mit Hilfe der gleichen Verbindung Signalisierungsnachrichten austauschen. Die Verbindung wird dabei zufällig von einem der beiden beteiligten Knoten gestartet.

3.6.1.2 Allgemeines Nachrichtenformat

Wie zuvor erläutert benötigt die Kontrollebene für ihre drei Signalisierungsprotokolle ein Transportprotokoll zur Absicherung gegen Datenfehler und Vertauschung von Paketen. Zu diesem Zweck werden

[22] Dies ist vergleichbar mit der Unterteilung in Knoten und Anwendungsadressen in heutigen Netzwerken. Dabei wird eine Paketzustellung zuerst anhand der Ziel-IP und anschließend, bei Empfang auf dem Zielknoten, auf Basis des Ziel-Ports der jeweiligen Anwendung durchgeführt.

die Mechanismen von TCP für den Transport der Signalisierungsnachrichten eingesetzt. Sie werden pro Verbindung angewandt, sodass Nachrichten mehrere Kanäle im Block (ähnlich zum Algorithmus von *Nagle* [107] bei TCP) versandt und ebenfalls im Block gegenüber dem Sender bestätigt werden (ähnlich zur Verwendung von *acknowledgement numbers* [108] bei TCP) können. Zur Bereitstellung der gewünschten Funktionen müssen TCP-spezifische Daten innerhalb der übertragenen Pakete gespeichert werden.

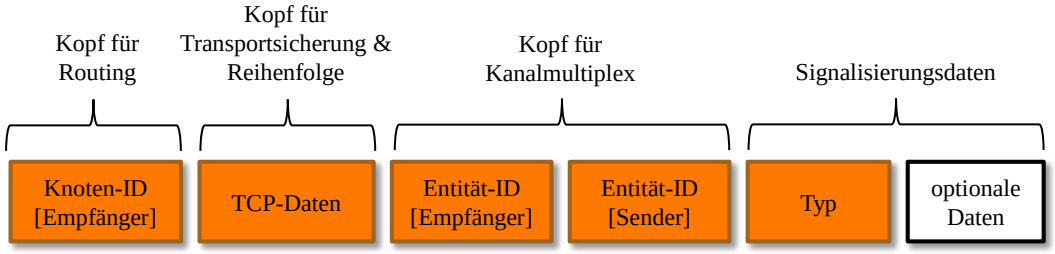

Abbildung 3.39: Paketaufbau zur Signalisierung innerhalb der Kontrollebene

Abbildung 3.39 zeigt den resultierenden Aufbau eines Pakets des Transportprotokolls von HRM. Dabei werden drei Kopfabschnitte verwendet:

- **Kopf für Routing:** Entsprechend Abschnitt 3.3.1 wird der Zielknoten eindeutig über seine Knoten-ID identifiziert. Diese ID ist ebenfalls für Routing notwendig, sodass ein Zwischenknoten der Route den jeweils nächsten Knoten in Richtung des Ziels bestimmen kann.
- **Kopf für Transportsicherung und Reihenfolge:** An dieser Stelle werden die TCP-spezifischen Daten zur Verhinderung von Dateninkonsistenz und Permutation der Paketreihenfolge-gespeichert. Dazu zählen Prüfsummen und Sequenznummern.
- **Kanalmultiplex:** Bei Punkt-zu-Punkt Signalisierungen der Kontrollebene werden Nachrichten stets zwischen genau zwei Entitäten ausgetauscht. Dabei sind sowohl Koordinatoren als auch Clustermanager an beiden Enden der Kommunikation möglich. Hierfür muss die Adressierung der Kontrollebene aus Abschnitt 3.3.3 auf Basis von Entität-IDs unterstützt verwenden. Sowohl die sendende als auch die empfangende Kontrollinstanz muss für den Empfangsknoten eindeutig gegeben sein, sodass er die Nachricht dem richtigen lokalen Kommunikationskanal zuordnen kann.

Es ist zu erkennen, dass auf eine Längenangabe im Paketkopf verzichtet wird. Stattdessen muss diese Information aus dem zugrundliegenden Protokoll gewonnen werden[23]. Im Anschluss an die drei Kopfbereiche folgt die eigentliche Signalisierungsnachricht. Sie beginnt mit einem Typenfeld, das die nachfolgenden Signalisierungsdaten klassifiziert und implizit die Gesamtlänge des Pakets festlegt. Nach dem Typenfeld folgen optionale Signalisierungsdaten des jeweiligen Signalisierungstyps. Eine Übersicht aller verwendeten Formate sowie den verwendeten Feldgrößen ist im Anhang B zu finden.

3.6.1.3 Spezielles Nachrichtenformat für Koordinatorwahlen

Während einer Koordinatorwahl werden Signalisierungen verschiedenen Typs verwendet. Die Priorität stellt die wichtigste Größe während einer Koordinatorwahl dar. Da sie veränderlich ist und sie sich durch gleichzeitige Wahlvorgänge schnell ändern kann, sollte sie mit möglichst kurzer Verzögerung an umliegende Knoten übermittelt werden. Dies unterstützt eine schnelle Konvergenz der Wahlvorgänge. Somit ist die Priorität des jeweiligen Senders in jeder Signalisierungsnachricht enthalten.

[23] Beispielsweise wird beim Einsatz von *Ethernet Frames* die Längenangabe durch die Implementierung des Protokolls von Schicht 2 gewonnen. Sie erkennt das Ende eines *Frames* mit Hilfe des *Interpacket Gap*, wodurch eine minimale Wartezeit nach dem Senden eines Frames definiert wird. Weitere Details dazu sind den Abschnitten 4.2.3.2.2 und 4.4.2 in [22] zu entnehmen.

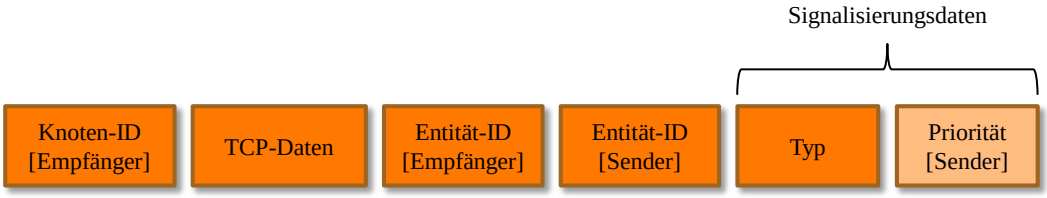

Abbildung 3.40: Paketaufbau zur Signalisierung innerhalb der Kontrollebene zur Koordinatorwahl

Abbildung 3.40 zeigt den resultierenden Aufbau von Signalisierungsnachrichten zur Koordinatorenwahl. Es ist zu erkennen, dass die Priorität des Senders als zusätzliches Datum in jeder Signalisierung übertragen wird.

3.6.2 Punkt-zu-Mehrpunkt-Signalisierungen

Diese Form der Signalisierung kommt bei *AnnounceNeighborNode*-Signalisierungen zur Erkennung von Nachbarknoten und bei der Bekanntgabe von Koordinatoren durch *AnnounceCoordinator*-Nachrichten zum Einsatz. Beide werden im Folgenden genauer erläutert.

3.6.2.1 Bekanntgabe von Nachbarknoten

In Abschnitt 3.3.1 werden *AnnounceNeighborNode* zur Erkennung von direkten Nachbarknoten eingeführt. Des Weiteren werden die Nachrichten in Abschnitt 3.5.1 zur Bestimmung der jeweiligen Verzögerung vorhandener Links eingesetzt.

Abbildung 3.41: Paketaufbau zur Bekanntgabe eines Nachbarknotens

In Abbildung 3.41 sind die beiden notwendigen Elemente einer *AnnounceNeighborNode* zur Umsetzung der beschriebenen Funktionen dargestellt:

- **Knoten-ID des Senders:** Der Quellknoten wird eindeutig über seine Knoten-ID identifiziert.
- **Anfrage-ID:** Mit Hilfe dieser ID ist eine Zuordnung zwischen einer ursprünglich gesendeten Anfrage zur Bekanntgabe und ihren nachfolgend eintreffenden Antworten möglich. Dies ist beispielsweise für die Bestimmung der Verzögerungszeit des jeweiligen Links zum Nachbarknoten notwendig.
- **Anfrage/Antwort-Marker**: Zwischen einer Anfrage und den darauffolgenden Antworten kann mit Hilfe dieses Markers unterschieden werden. Dabei beinhaltet eine Antwortnachricht die ID ihrer zugehörigen Anfragenachricht.

3.6.2.2 Bekanntgabe von Koordinatoren

Entsprechend Abschnitt 3.3.5 werden *AnnounceCoordinator*-Nachrichten eines Koordinators konzentrisch im Netzwerk ausgebreitet. Die Ausbreitung ist durch den Radius *r* automatisch begrenzt. Neben der Identifikation des bekanntgegebenen Koordinators müssen dabei ebenfalls die notwendigen Routingdaten mitgeteilt werden. Weitere Details dazu wurden in Abschnitt 3.3.3 beschrieben.

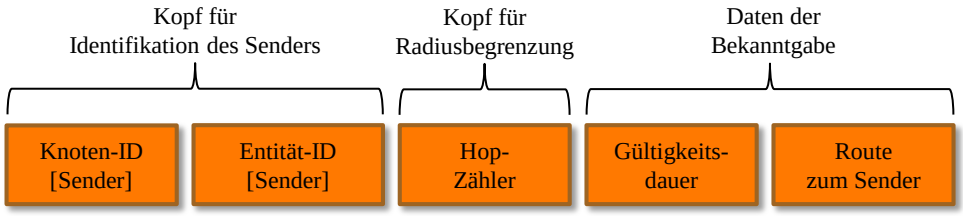

Abbildung 3.42: Paketaufbau zur Bekanntgabe eines Koordinators

Abbildung 3.42 zeigt das resultierende Format für *AnnounceCoordinator*-Nachrichten. Darin sind zwei Bereiche zu erkennen: Kopf- und Datenteil. Der Kopfteil besteht aus folgenden Teilen:

- **Identifikation des Senders:** Der Sender muss eindeutig identifiziert werden, sodass ein Empfänger einer solchen Nachricht ihn eindeutig zuordnen kann. Eine Knoten-ID identifiziert den sendenden Knoten, die sendende Koordinatorinstanz wird über die Entität-ID des zugehörigen Clustermanagers beschrieben.
- **Radiusbegrenzung:** Ein Zähler gibt die Anzahl von Hops an, welche (bezogen auf das jeweilige Hierarchielevel des sendenden Koordinators) bereits passiert worden sind. Auf Basis dieses Wertes prüft ein Empfänger entsprechend Abschnitt 3.3.5, ob der maximal erlaubte Radius *r* für die Ausbreitung der Bekanntgabe bereits erreicht ist und stoppt in diesem Fall die Weiterleitung der Nachricht[24].

Der hinter dem Kopfteil gelagerte Datenteil bietet sowohl Information über die Gültigkeitsdauer der Bekanntgabe als auch die notwendigen Routingdaten zum Aufbau neuer Kommunikationen innerhalb der Kontrollebene. Folgende Elemente beinhaltet der Datenteil:

- **Gültigkeitsdauer:** Dieser skalare Wert gibt in Sekunden die Gültigkeitsdauer dieser Nachricht an. Der Empfänger verwendet sie, um lokal den Zeitpunkt zu berechnen, ab dem er den bekannt gegebenen Koordinator wiederum als ungültig annehmen muss. Nur durch periodische Bekanntgabe eines Koordinators, gilt dieser für alle umliegenden Knoten als dauerhaft verfügbar und eine aufgebaute Kommunikation bleibt bestehen. Auf Basis dieses Mechanismus sind sowohl Knotenausfälle als auch Koordinatorabwahlen aufgrund von Topologieänderungen für umliegende Knoten erkennbar.
- **Route zum Sender:** Diese Liste mit eindeutigen Knoten-IDs verwendet jeder Empfänger, um den nächsten Nachbarknoten in Richtung des Senders zu bestimmen.
  Des Weiteren wird aus der Route abgeleitet, ob die Nachricht bei Weiterleitung an einen Nachbarn eine Schleife im Netzwerk durchläuft. In diesem Fall ist seine Knoten-ID bereits in der Liste enthalten und eine Weiterleitung wird verhindert.
  Aus der Menge enthaltener Knoten-IDs ergibt sich ebenfalls die Länge der Route zum Sender. Der Empfänger kann entscheiden, ob die übermittelte Route zum Sender länger als die bisher bekannte ist. Ist dies der Fall, kann die empfangene Route ignoriert werden. Entspricht sie jedoch einer kürzeren, wird sie zukünftig für die Signalisierung innerhalb der Kontrollebene genutzt.

[24] Für eine Hierarchietiefe ab 4 muss zusätzlich die Entität-ID des zuletzt passierten Clusters (bezogen auf das Hierarchielevel des sendenden Koordinators) an dieser Stelle gespeichert werden, nähere Details dazu sind in Abschnitt 3.3.5.1 zu finden. Im Vordergrund dieser Arbeit steht jedoch eine Hierarchietiefe von 3.

Im Gegensatz zu allen anderen Nachrichtentypen werden *AnnounceCoordinator*-Nachrichten im Netzwerk über mehrere Knoten weitergeleitet. Jeder Empfänger muss dabei den Datenteil aktualisieren, indem er seine eigene Knoten-ID der Route zum Sender hinzufügt. Somit gilt er für den nächsten Empfänger als möglicher nächster Router auf dem Weg zum Sender.

3.6.3 Einordnung im OSI-Modell

Eine Nachbarschaftsbeziehung besteht im Kontext von HRM stets auf Basis vorhandener Broadcast-Domänen. Folglich ist ein Nachbar ein Knoten, welcher über eine Broadcast-Nachricht des Protokolls von Schicht 2 erreicht werden kann. Beispielsweise können die Signalisierungsnachrichten der Kontrollebene auf Basis von *Ethernet Frames* und dem damit verbundenen Adressierungsschema übertragen werden. Nähere Details sind in Anhang B.4 zu finden.

Des Weiteren ist es möglich, die Signalisierungen mit Hilfe eines vorhandenen Protokolls von Schicht 3 und darüber zu realisieren, insofern es die zuvor genannten Annahmen aus Abschnitt 3.3.9 erfüllt. Weitere Details dazu werden in den Implementierungsbeschreibungen von Abschnitt 4.2.4 gegeben. Aufgrund der zuvor genannten Abhängigkeiten und Möglichkeiten sind die Protokolle der Kontrollebene innerhalb des OSI-Modells auf Schicht 3 und darüber einzuordnen.

3.7 Routingmanager

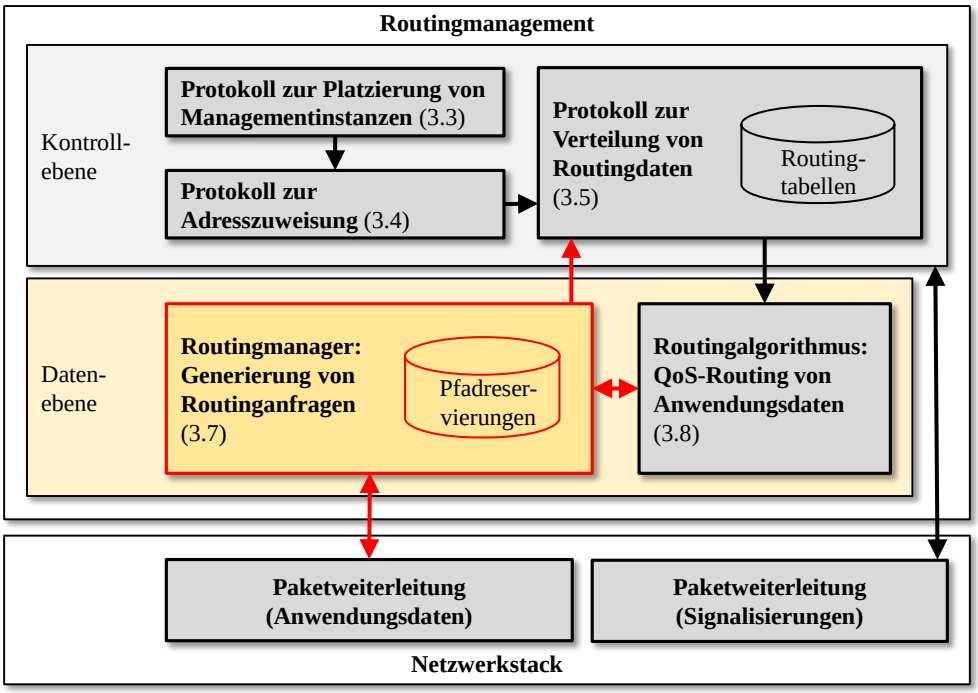

Abbildung 3.43: Position und Interaktion des Routingmanagers innerhalb der HRM-Architektur

Wie in Abbildung 3.43 zu erkennen ist, besteht die Datenebene aus zwei funktionalen Komponenten: Sie beinhaltet neben dem allgemeinen Routingalgorithmus einen Routingmanager auf jedem Knoten. Jede dieser Instanzen interagiert mit der durch das Protokoll von Schicht 3 bereitgestellten Paketweiterleitung. Dabei leitet er die für die lokale Paketverarbeitung notwendigen Routinganfragen ab und leitet diese an den Routingalgorithmus weiter. Dessen ermittelte Routingentscheidung wird wiederum in umgekehrter Richtung an die Paketweiterleitung zurückübermittelt, sodass das Paket an den jeweils nächsten Knoten in Richtung seines Ziels weitergeleitet wird. Der Routingmanager stellt somit auf einem

Knoten die funktionale Verbindung zwischen der Paketweiterleitung und dem lokal ablaufenden Routingalgorithmus dar[25]. Für jedes eintreffendes Paket führt er die folgenden Einzelschritte in der aufgeführten Reihenfolge aus:

1. **Ermittlung von Qualitätsanforderungen:** Der Routingalgorithmus benötigt neben der lokalen Routingtabelle, welche durch die Kontrollebene für jeden Knoten erstellt wird, ebenfalls die Qualitätsanforderungen der Anwendung. Diese Daten können im Allgemeinen entweder per *Inband*- oder *Outband*-Signalisierung am Knoten eintreffen. Dabei ist durch HRM nicht spezifiziert, welches QoS-Modell und welches Signalisierungsprotokoll verwendet werden muss, diese Entscheidungen können zwischen verschiedenen Netzwerkstacks und Anwendungsszenarien von HRM variieren.
   Bei ***Inband***-Signalisierungen werden die Anforderungen in den Paketen des Datenstroms kodiert, typischerweise erfolgt dies auf dem Quellknoten. Die Knoten im Netzwerk speichern in dem Fall keine Zustandsdaten, in welchen die geforderten Qualitätsmerkmale für die Übertragung enthalten sind. Für die Routingentscheidungen auf den nachfolgenden Knoten werden ausschließlich die Metadaten aus den eintreffenden Paketen ausgewertet. Beispielsweise können dabei die Qualitätsanforderungen der Anwendung in jedem IP-Paket als optionale Daten im Paketkopf kodiert sein.
   Im Gegensatz zur *Inband*-Signalisierung wird bei der ***Outband***-Signalisierung ein separates Protokoll für die Übermittlung von Qualitätsanforderungen verwendet. Beispiele eines solchen Protokolls sind RSVP oder NSIS, beide dienen zur Signalisierung von Qualitätsanforderungen der Anwendung und nachfolgender Reservierung eines festen Pfads durch das Netzwerk. Dieser festgelegte Weg wird nachfolgend für die Übertragung der einzelnen Pakete des jeweiligen Datenstroms verwendet.
   Während das Konzept von HRM grundsätzlich für beide genannten Methoden zur Anwendung kommen kann, liegt innerhalb dieser Arbeit der Fokus auf der Outband-Signalisierung und der Verwendung von festen Ressourcenreservierungen für jeden Datenstrom entsprechend dem *IntServ*-Modell.
2. **Ermittlung einer Routingentscheidung:** Falls das Paket zu keiner bereits vorhandenen Pfadreservierung zugeordnet werden kann oder *DiffServ* verwendet wird, muss eine Anfrage an den Routingalgorithmus gestellt werden. Er ermittelt den nächsten Knoten in Richtung des jeweiligen Paketziels unter Beachtung der als Parameter der Anfrage übergebenen Qualitätsanforderungen und der aktuell bekannten Eigenschaften von existierenden Routen im Netzwerk, letztere sind in der jeweils lokal gespeicherten Routingtabelle beschrieben.
3. **Pfadreservierungen:** Wird das *IntServ*-Modell eingesetzt, kommen typischerweise Pfadreservierungen zum Einsatz. Fall das jeweilige Paket zu einem noch unbekannten Datenstrom gehört, werden durch den Routingmanager eine entsprechende Reservierung von Ressourcen für den jeweils ausgehenden Link in Richtung des nächsten Knotens veranlasst.
4. **Aktualisierung von Paketdaten:** Der Routingmanager hat die Aufgabe, etwaige im Paket vorhandene Qualitätsanforderungen für die Verzögerung zu aktualisieren. Da sich durch die Nutzung der jeweils ausgewählten nächsten Netzwerkschnittstelle, und des sich dahinter befindlichen Links, die maximal erlaubte Verzögerung für den restlichen Teil der Übertragungsstrecke zum Ziel verringert, muss der im Paket signalisierte Wert entsprechend verringert werden.
5. **Aktualisierung von Routingdaten:** Sollte eine neue Pfadreservierung erstellt worden sein, informiert der Routingmanager die Kontrollebene über das Ereignis. Diese löst neue Signalisierungen des Protokolls zur Verteilung von Routingdaten aus. Dazu genügt es, wenn die lokal

[25] Diese Form der Abstraktion wird insbesondere verwendet, um HRM allgemein beschreiben zu können. Die implementierungsspezifischen Eigenschaften des jeweils verwendeten Netzwerkstacks werden dabei durch den Manager gegenüber dem Routingalgorithmus gekapselt, sodass das in diesem Kapitel vorgestellte Konzept sowohl für IP als auch für alternative Lösungen, wie beispielsweise FoG, eingesetzt werden kann.

gespeicherten Routingdaten aktualisiert werden. Die Veränderungen werden automatisch erkannt und über *RouteReport*-Nachrichten an übergeordnete Koordinatoren gemeldet, sodass dadurch ebenfalls eine Aktualisierung der Inhalte von *RouteShare*-Nachrichten ausgelöst wird. Folglich werden dadurch die Einträge in Routingtabellen von entfernten Knoten entsprechend den veränderten Eigenschaften von vorhandenen Routen angepasst.

3.8 Routingalgorithmus

Die zweite funktionale Komponente der Datenebene stellt der eigentliche Routingalgorithmus dar, der auf Anfrage des Routingmanagers eine Routingentscheidung für Anwendungspakete unter Berücksichtigung der Qualitätsanforderungen der jeweiligen Anwendung ermittelt. Er wird auf allen Knoten in gleicher Form angewendet, jede von ihm ermittelte Routingentscheidung ist unabhängig von vorhergehenden Entscheidungen auf anderen Knoten.

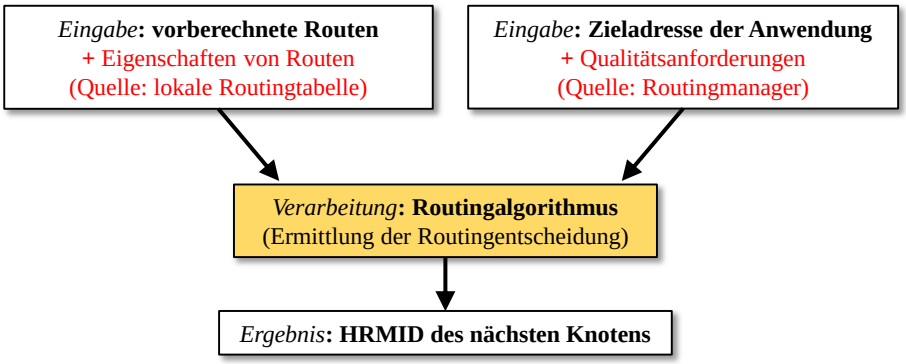

Abbildung 3.44: Prinzipieller Datenfluss zur Ermittlung einer Routingentscheidung

Abbildung 3.44 gibt eine Übersicht über den Datenfluss zur Ermittlung einer Routingentscheidung, der Routingalgorithmus verwendet dabei zwei Eingaben:

- **Vorberechnete Routen:** Als erste Eingabe sind auf der linken Seite die vorberechneten Routen aufgeführt, welche auf jedem Knoten in Form der lokal abgespeicherten Routingtabelle vorliegen. Sie beschreibt die aktuellen IST-Werte von bekannten Routen, dazu zählen auch die aktuell bekannten Eigenschaften jeder Route. Ihre Einträge sind nach erfolgreichem Aufbau der Kontrollebene sowie der Verteilung von Knotenadressen und Signalisierung von Routingdaten vorhanden. Sollte eine Routinganfrage vor Erreichen dieses Zustandes auftreten, ist das durch den Routingalgorithmus ermittelte Ergebnis von der Menge von bereits bekannten Routen abhängig. Durch die periodischen Signalisierungen der Kontrollebene werden die Einträge jeder Tabelle kontinuierlich aktualisiert, sodass eine Routingentscheidung immer auf Basis von möglichst aktuellen Routingdaten getroffen wird – es wird ein dynamisches Routing angewandt.
- **Zieladresse der Anwendung:** Als zweite Eingabe ist auf der rechten Seite die Zieladresse aufgeführt. Diese wird durch den Routingmanager zusammen mit den Qualitätsanforderungen der jeweiligen Anwendung an den Routingalgorithmus als sogenannte SOLL-Werte für die gewünschte Route übergeben.

Als Ergebnis einer erfolgreichen Routinganfrage erhält der Routingmanager eine HRMID des Nachbarknotens, der als nächster in Richtung des Zielknotens der Anwendungsdaten verwendet werden muss.

Bei der Betrachtung von Abbildung 3.44 ist es wichtig, den Unterschied zwischen Routenberechnungen und Routingentscheidungen zu beachten. Während erstere auf Basis der durch die Kontrollebene signalisierten Routingdaten die Einträge in den knotenlokalen Routingtabellen ermitteln, werden die tatsächlichen Entscheidungen für die Wegewahl durch letztere getroffen. Das dabei angewandte Vorgehen

stellt den Routingalgorithmus dar, dieser wird in den nachfolgenden Abschnitten detailliert erläutert. Details zur durchgeführten Implementierung des Algorithmus sind in Abschnitt 4.3.4 zu finden.

3.8.1 Anforderungen an den Routingalgorithmus

Das Konzept eines Routingalgorithmus verfolgt stets Designziele und orientiert sich an aufgestellten Anforderungen. Entsprechend den in Abschnitt 2.2.3.4 aufgestellten Anforderungen an einen Routingalgorithmus muss das durch HRM bereitgestellte QoS-Routing sowohl Datenrate als auch Verzögerung einzelner Links beachten. Als Gegenstück dazu müssen bei Routingentscheidungen insbesondere Qualitätsanforderungen der Anwendung beachtet werden:

- **Benötigte Datenrate:** Die Anwendung kann eine notwendige Datenrate festlegen, welche entlang der ermittelten Route möglich sein soll, sodass ihre Daten mit der gesendeten Geschwindigkeit übertragen und ohne Verluste am Empfänger eintreffen.
- **Erlaubte Verzögerung:** Neben einer geforderten Datenrate kann ebenfalls eine maximale Verzögerung für die Übertragung gegeben sein, welche entlang der Gesamtroute nicht überschritten werden darf. Dies ist insbesondere im Kontext von Videokonferenzen interessant, um Routen mit sehr hoher Verzögerung möglichst zu vermeiden.

Neben den zuvor genannten QoS-spezifischen Anforderungen muss der Algorithmus von HRM zusätzlich allgemeinen Anforderungen an das resultierende QoS-Routing gerecht werden:

- **Determinismus:** Der Algorithmus muss auf Basis gleicher Eingabedaten stets die gleiche Routingentscheidung als Ergebnis liefern.
- **Zuverlässigkeit:** Unabhängig von der aktuell vorliegenden Routingtabelle und der gegebenen Qualitätsanforderungen muss der Algorithmus ein Ergebnis in endlicher Zeit liefern.
- **Verhinderung von Überlast:** Der Routingalgorithmus vermeidet proaktiv die vollständige Sättigung einzelner Links.
- **Fairness:** Ein Datenstrom sollte möglichst wenige parallel ablaufende Übertragungen beeinflussen.
- **Robustheit:** Der Algorithmus muss Topologieänderungen verarbeiten und bei Ausfällen durch Rerouting reagieren.
- **Eigenständigkeit:** Der Algorithmus darf keine manuellen Eingaben benötigen.

Auf Basis der gegebenen Anforderungen an die Konzeption wird der Routingalgorithmus in den folgenden Abschnitten genauer spezifiziert.

3.8.2 Ermittlung einer Routingentscheidung

In Abschnitt 2.2.4 wurden die grundlegenden Routingstrategien SWPF, WSPF und BFF beschrieben. Im Folgenden wird ihre Anwendung diskutiert, um die für HRM ausgewählte Lösung zu begründen. Dabei liegt der Fokus auf einer Erbringung eines dynamischen Routings, welches die jeweils aktuellen Routingdaten beachtet.

3.8.2.1 Differenzierung zwischen möglichen Varianten

Da es aufgrund der Laufzeitverzögerungen im Netzwerk denkbar ist, dass sich die Eigenschaften einer Route ändern, ist eine Anwendung der BFF-Strategie allgemein kritisch zu sehen. Schnell können die bei einer Routingentscheidung als „gerade noch ausreichend“ eingestuften QoS-Eigenschaften während der Weiterleitung entlang der gewählten Route auf einem Zwischenknoten plötzlich nicht mehr verfügbar sein. Parallel hinzukommende Reservierungen für neue Datenströme können dies verursachen.

Des Weiteren spielt die Vermeidung von Linksättigungen eine wichtige Rolle für die Gesamtperformanz des Routings. Die Routingstrategie sollte eine vollständige Auslastung einzelner Routen, und somit auch

von Links, vermeiden. Insbesondere ist die bisher bekannte WSPF-Strategie vor diesem Hintergrund kritisch zu sehen, da sie die verfügbare Auslastung einzelner Routen nicht beachtet und sich primär an den Routenlängen orientiert.

Wird aufgrund der vorhergehenden Überlegungen ausschließlich die alternative SWPF-Routingstrategie angewandt, spielen die aktuell verfügbaren Datenraten entlang der bekannten Routen die vordergründige Rolle. Diese Strategie erzielt für eine einzelne Anwendung sehr gute Werte, da stets die leistungsstärkste Route ausgewählt und die Überlastung von Links weitestgehend verhindert wird. Jedoch wirkt sich dieses Vorgehen negativ auf die in Abschnitt 3.8.1 geforderte Fairness bei der Zuweisung von Netzwerkressourcen aus.

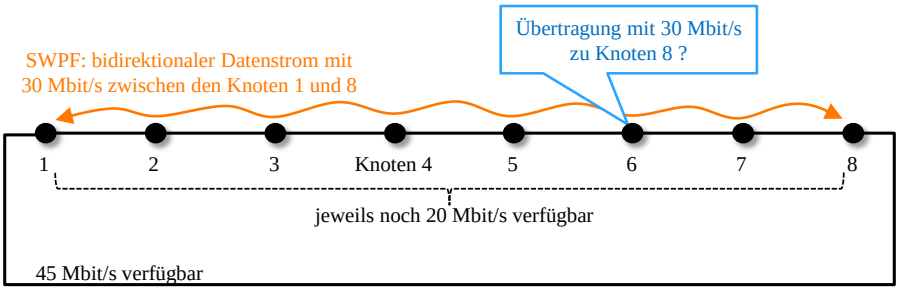

Abbildung 3.45: ein Datenstrom verhindert das erfolgreiche Routing eines neuen Datenstroms

Abbildung 3.45 zeigt ein Beispiel für schlechte Routingfairness, wenn ausschließlich die SWPF-Strategie eingesetzt wird. Der orange dargestellte Datenstrom wird entsprechend SWPF entlang der längeren Route mit der zuvor höchsten verfügbaren Datenrate von 50 Mbit/s geleitet. Im Vergleich zu WSPF erhält er die Ressourcen von signifikant mehr Links und kann dadurch parallele Routinganfragen auf den betroffenen Zwischenknoten blockieren, als Beispiel ist die blau dargestellte Routinganfrage auf Knoten 6 zu sehen.

3.8.2.2 Gewählte Lösung

Die für HRM gewählte Routingstrategie vermeidet BFF und setzt stattdessen auf eine Kombination aus WSPF mit SWPF.

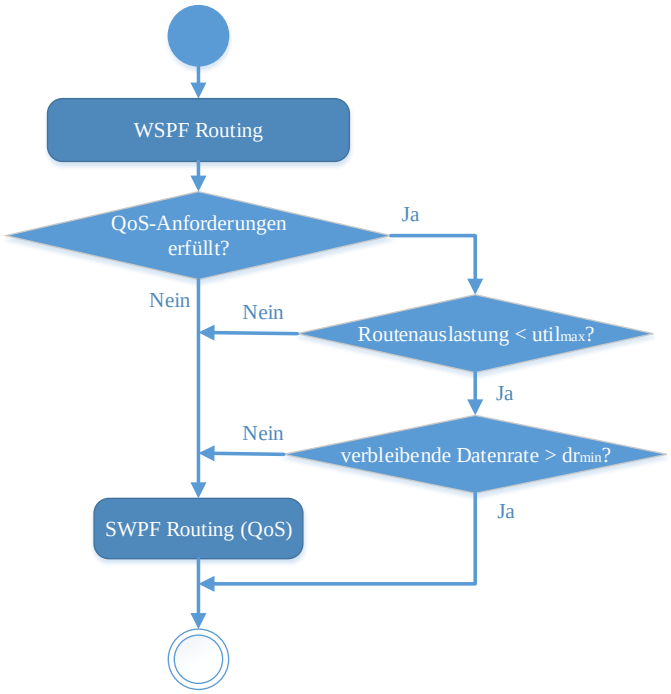

Abbildung 3.46: Ermittlung einer Routingentscheidung

Abbildung 3.46 beschreibt den resultierenden Routingalgorithmus als Diagramm. Im ersten Schritt wird die beste Route für die WSPF-Strategie ermittelt. Im nächsten Schritt wird geprüft, ob die geforderten QoS-Eigenschaften durch die gefundene Route erfüllt sind. Sind durch die Anwendung keine Qualitätsanforderungen explizit gegeben, gelten die Anforderungen als erfüllt und der Routingalgorithmus wendet einfaches BE-Routing an. Sollte das Ergebnis von WSPF bereits die Anforderungen erfüllen, werden zusätzlich die folgenden zwei Bedingungen geprüft:

- **Routenauslastung < $util_{max}$:** Zusätzlich zu den bisher genannten QoS-Eigenschaften (Datenrate und Verzögerung) wird die Auslastung in die Routingentscheidung einbezogen. Sie wird über die Kontrollebene unter den Netzwerkknoten signalisiert. Diese Bedingung dient der Vermeidung überlasteter Routen. Ist die gefundene kürzeste Route zum Ziel überlastet, wird die Lösung der SWPF-Strategie verwendet. Der genaue Wert von $util_{max}$ wird dabei durch die jeweilige Implementierung bestimmt.
- **Verbleibende Datenrate > dr_{min}:** Ziel dieser Bedingung ist es, Links mit sehr geringer Kapazität unabhängig ihrer Auslastung zu meiden. So kann beispielsweise eine stärker genutzte Route, bestehend aus mehreren 100 Mbit/s Links, gegenüber einem bisher ungenutzten DSL-Uplink mit 128kbit/s maximaler Datenrate im Routing vorgezogen werden, obwohl die DSL-Verbindung die kürzere Route darstellt. HRM gibt keinen expliziten Wert für dr_{min} vor, da ein optimaler Wert von der Anwendung und dem verwendeten Netzwerk abhängt.

Sollte eine der genannten Bedingungen nicht zutreffen, wird zur Lösung von SWPF-Routing gewechselt. Der Routingalgorithmus wählt in diesem Fall die Route mit der größten verfügbare Datenrate aus. Als untergeordnete Entscheidungskriterien dienen die Verzögerung sowie die Routenlänge. Durch die Fokussierung auf die Datenrate wird ebenfalls eine automatische Lastverteilung zwischen den bekannten Routen realisiert, sodass für SWPF die Auslastung einzelner Routen nicht geprüft wird.

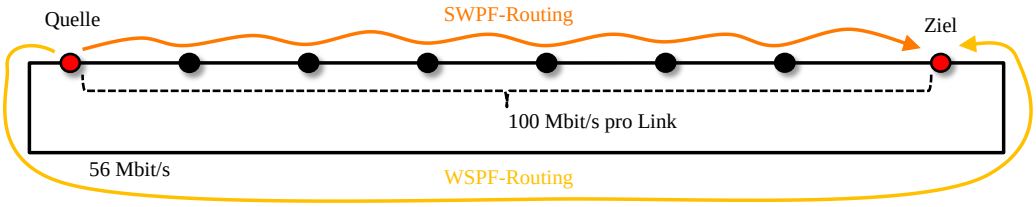

Abbildung 3.47: Routing über WSPF-Pfade (gelb) und bei Überlastung folgt Nutzung von SWPF-Pfaden (orange)

Abbildung 3.47 zeigt die Umschaltung zwischen der WSPF- zur SWPF-Route für das Referenzszenario. Der Quellknoten sendet in dem Fall kontinuierlich immer mehr Datenströme zum Zielknoten. Diese werden entlang der gelben Route geleitet. Unter der Annahme, dass die Auslastung den Wert $util_{max}$ = 80 % möglichst nicht übersteigen soll, wechselt das Routing sobald ein zusätzlicher Datenstrom die Gesamtauslastung eines verwendeten Links auf einen Wert größer 44,8 Mbit/s – und somit größer 80% – erhöht. Für die weiteren Datenströme wird nachfolgend die SWPF-Strategie eingesetzt, sodass die orange dargestellte Route zum Einsatz kommt. Da die SWPF-Strategie aber stets die Route mit der höchsten Datenrate verwendet, bieten die verbleibenden 20 % der Ressourcen entlang der gelben Route bei hoher Auslastung der orangen Route irgendwann wiederum mehr verfügbare Datenrate. Folglich werden die verfügbaren Kapazitäten aller Routen nach und nach reserviert. Sollte irgendwann keine Route die Qualitätsanforderungen erfüllen, kann die Implementierung weiterhin die Route mit den bestmöglichen Eigenschaften als Ergebnis liefern oder die Routinganfrage direkt ablehnen.

3.8.3 Kostenmodelle für WSPF- und SWPF-Routing

Zur Ermittlung einer Routingentscheidung werden bei HRM sowohl für die WSPF- als auch die SWPF-Strategie die Eigenschaften von bekannten Routen durch ein geeignetes Kostenmodell (Routenmetrik) in einen Skalarwert überführt. Er dient zur quantitativen Bewertung von einzelnen Routen und ordnet ihnen Kostenwerte zu, ein höherer Wert steht somit jeweils stellvertretend für höhere Kosten bei Verwendung der jeweiligen Route und beschreibt eine schlechtere Routenwahl. Das Kostenmodell beachtet folgende Werte einer Route:

- $\boldsymbol{dr_{Route}}$**:** Die Datenrate dr_{Route} besitzt umgekehrt proportionalen Einfluss auf die Gesamtkosten. Ihre verursachten Kosten werden durch 1 / dr_{Route} beschrieben. Routen mit höhere Datenraten werden somit bevorzugt behandelt.
- $\boldsymbol{del_{Route}}$**:** Als zweiter QoS-spezifischer Parameter existiert die Verzögerung einer Route, die Gesamtkosten verhalten sich dazu proportional. Dadurch sind Routen mit höherer Übertragungsverzögerung von geringerer Bedeutung.
- $\boldsymbol{hc_{Route}}$**:** Als dritter Parameter existiert die Routenlänge. Die Gesamtkosten sind proportional zur Routenlänge, sodass die kürzeste Route Vorrang besitzt.

Die genaue Berechnung der Kosten einer Route ist abhängig davon, mit welcher Priorisierung die oben aufgeführten Parameter verwendet werden. Dies wird in den nachfolgenden Abschnitten genauer beschrieben. Weitere Details über die innerhalb der Implementierung enthaltene Umsetzung der nachfolgend erläuterten Kostenmodelle sind in Abschnitt 4.3.4 zu finden.

3.8.3.1 Priorisierung zwischen Datenraten und Verzögerung

Der Routingalgorithmus kann für eine zu treffende Entscheidung versuchen, ein optimales Ergebnis sowohl in Bezug auf Datenrate als Verzögerung zu finden. Aus [63] und [24] geht jedoch hervor, dass eine solche Strategie ein NP-vollständiges Problem darstellt und somit hohe Berechnungszeiten verursacht. Darüber hinaus sind Szenarien denkbar, in denen die Forderung nach optimaler Datenrate und Verzögerung aufgrund der Kapazitätsverteilung im Netzwerk nicht gleichzeitig eingehalten werden können. Zur Lösung wird in dieser Arbeit zwischen beiden Anforderungen eine Priorisierung für den

Routingalgorithmus eingeführt, um sowohl die Laufzeit zu reduzieren als auch einen möglichen Konflikt der Zielstellungen zu vermeiden.

HRM kann prinzipiell für alle Typen von Netzwerken eingesetzt werden. Die in Kapitel 6 dargestellten Untersuchungen sind jedoch auf Firmen- und Providernetzwerke fokussiert, sodass aufgrund der eher geringen geographischen Distanzen die Verzögerungsunterschiede zwischen vorhandenen Routen gegenüber der jeweils gebotenen Datenrate geringere Relevanz für die resultierende Übertragungsqualität aufweisen. Folglich wird in dieser Arbeit die Datenrate gegenüber der Verzögerung höher priorisiert, wodurch ebenfalls die Vermeidung von Paketverlusten unterstützt wird.

3.8.3.2 Routenkosten für WSPF-Routing

Auf Basis der zuvor beschriebenen Priorisierung kann im Folgenden eine Kostenformel festgelegt werden.

$$c_{WSPF} = del_{Route} + (1 / dr_{Route} + hc_{Route} \cdot W_2) \cdot W_1$$

Formel 3.7: Berechnung der Routingkosten für WSPF-basiertes QoS-Routing

Formel 3.7 zeigt die Berechnung der Gesamtkosten c_{WSPF} für die WSPF-Strategie. Die enthaltenen Gewichte W_1 und W_2 setzen die zuvor angesprochene Priorisierung zwischen Datenrate und Verzögerung um. Des Weiteren ermöglichen sie der Routenlänge als primären Parameter den größten Einfluss auf die Gesamtkosten. Die verwendeten Gewichte müssen hierfür stets größer 1 sein.

$$W_1 \geq \text{max. Wert für } del_{Route}$$
$$W_2 \geq \text{max. Wert für } dr_{Route}$$

Abbildung 3.48: Bedingungen zur Berechnung der Routingkosten der WSPF-Strategie

Zusätzlich muss es in einer realen Implementierung Annahmen über Maximalwerte für Datenrate und Verzögerung geben, sodass sinnvolle Gewichte explizit festgelegt werden können. Abbildung 3.48 verdeutlicht dies. Erst dadurch wird sichergestellt, dass die gewünschte Priorisierung für Formel 3.7 beachtet wird.

3.8.3.3 Routenkosten für SWPF-Routing

$$c_{QoS} = hc_{Route} + (del_{Route} + 1 / dr_{Route} \cdot W_3) \cdot W_4$$

Formel 3.8: Routingkosten für Routinganfragen mit Qualitätsanforderungen

Für SWPF-basiertes Routing werden die Routingkosten entsprechend Formel 3.8 berechnet. Auch hier sind Gewichte zur Abbildung der Prioritätsverteilung notwendig. Die Gewichte ordnen erneut der Datenrate im Vergleich zur Verzögerung eine höhere Wichtigkeit zu. Die Routenlänge wird als unwichtigste Größe für die Gesamtkosten einer Route betrachtet. Die verwendeten Gewichte müssen ähnlich Formel 3.8 ebenfalls größer 1 sein und die Bedingungen aus Abbildung 3.49 erfüllen.

$$W_3 \geq \text{max. Wert für } del_{Route}$$
$$W_4 \geq \text{max. Wert für } hc_{Route}$$

Abbildung 3.49: Bedingungen zur Berechnung der Routingkosten der SWPF-Strategie

Zur Bestimmung der Routingkosten der SWPF-Strategie wird die Priorisierung zwischen Datenrate und Verzögerung aus Abschnitt 3.8.3.1 verwendet und durch und Formel 3.8 umgesetzt. Dabei kann zur Vereinfachung das Gewicht W_3 dem Wert von Gewicht W_1 entsprechen.

3.8.4 Anwendung im *IntServ*-Modells

Im Rahmen dieser Arbeit steht insbesondere die Anwendung des Routingalgorithmus für das *IntServ*-Modell im Vordergrund. Die durch den Algorithmus ermittelte Entscheidung ist stets abhängig von der aktuell bekannten Lastsituation im Netzwerk. Jeder Routingmanager sorgt wiederum dafür, dass die für die Übertragung eines Datenstroms notwendigen Ressourcen automatisch reserviert werden und alle zugehörigen Pakete stets entlang des gleichen ausgehenden Links in Richtung des Ziels weitergeleitet werden. Somit besteht eine einmal festgelegte Route unabhängig der eventuell in den Routingtabellen auftretenden Veränderungen. Die Untersuchung von Möglichkeiten zur automatischen Neukonfiguration von bereits festgelegten Reservierungen (bspw. bei Ausfall von Routern oder bei lokalen Kapazitätsengpässen) ist Thema für zukünftige Forschungsarbeiten und nicht Bestandteil dieser Arbeit.

3.8.5 Anwendung im *DiffServ*-Modells

Das Konzept von HRM kann ebenfalls für das *Diffserv*-Modell angewandt werden. Analog zum IntSserv-Modell sind dabei die Entscheidungen des Routingalgorithmus abhängig von dem Inhalt der jeweils lokalen Routingtabelle, welche die aktuell bekannten Routen sowie ihre QoS-spezifischen Eigenschaften beinhaltet. Sollten sich während der Betriebszeit des Netzwerks die QoS-Eigenschaften einer Route drastisch verschlechtern, werden durch den Routingalgorithmus unter Beachtung der Qualitätsanforderungen der Anwendung automatisch alternative Wege im Rahmen der vorhandenen Möglichkeiten ausgewählt. Folglich werden auftretende Engpässe durch automatisches Rerouting kompensiert. Bei dieser Überlegung muss jedoch auch der schlechteste Fall beachtet werden: Schnell wiederkehrende Veränderungen in der Lastverteilung können im Netzwerk auftreten. Ohne weitere Maßnahmen kann dies die Stabilität von Routingentscheidungen signifikant negativ beeinflussen, sodass häufiges Rerouting auftritt. Dies verursacht zusätzliche Veränderungen in der Lastverteilung, wodurch häufige Signalisierungen der Kontrollebene zur Aktualisierung von Routingdaten auftreten. Diese Situation kann dazu führen, dass jeweils eine Aktualisierung entsprechend des in Abschnitt 3.5.4.2 beschriebenen Intervalls zwischen den beteiligten Entitäten der Kontrollebene ausgetauscht wird und das Signalisierungsaufkommen signifikant ansteigt. Als mögliche Gegenmaßnahme kann das Konzept von HRM beispielsweise mit einem Punktesystem erweitert werden, welches ähnlich zu *BGP route flap damping* [109] aufgebaut ist. Dabei werden häufig wiederkehrende Veränderungen in den Routingdaten mit einer hohen Punktezahl bewertet und als kritisch eingestuft. Sollte ein definierter Schwellwert für eine signalisierte Route erreicht werden, kann die Aktualisierung von betroffenen Routingdaten automatisch verlangsamt oder sogar verhindert werden. Dadurch wird die Stabilität der Einträge in Routingtabellen – und somit auch die Stabilität von Routingentscheidungen – verbessert. Folglich werden die Pakete eines Datenstroms eher entlang der gleichen Route durch das Netzwerk geleitet, wodurch auch die Menge von verursachten Signalisierungen der Kontrollebene gering gehalten wird. Die detaillierte Konzeption dieses Ansatzes sowie die möglichen Auswirkungen der Vorgehensweise müssen jedoch in zukünftigen Forschungsarbeiten näher untersucht werden. Im Rahmen dieser Arbeit ist der Fokus jedoch auf das *IntServ*-Modell gerichtet und alle weiteren Pakete werden nach der WSPF-Strategie (ähnlich zu reinem BE-Routing) durch das Netzwerk weitergeleitet.

3.9 Interoperabilität mit heutigen IPv4/IPv6

Für eine erfolgreiche Verbreitung der HRM-Signalisierung in realen Netzwerken ist eine Kompatibilität mit bisherigen IP-Netzen wichtig.

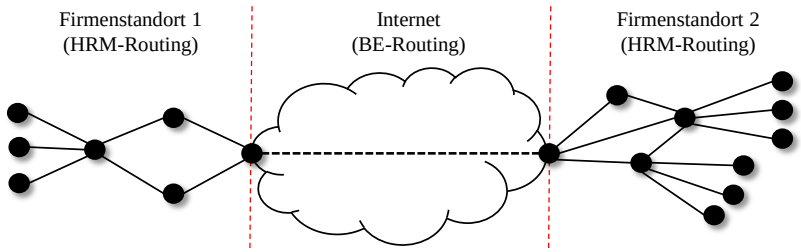

Abbildung 3.50: Interoperabilität zweier HRM-Netzwerke mit dem per BE-Routing arbeitenden Internet

Das in Abbildung 3.50 dargestellte hybride Netzwerk beinhaltet zwei HRM-Netzwerke von Firmenstandorten. Diese sind mit Hilfe des Internets miteinander verbunden. In einem solchen Szenario muss es möglich sein, dass Daten zwischen beiden Standorten ohne Anpassungen an den Routern des Internets übertragen werden können. Ebenfalls muss es möglich sein, dass Knoten eines HRM-Netzwerks mit Knoten im Internet kommunizieren können.

3.9.1 Bedingungen für eine Implementierung

Für einen Einsatz von HRM in IP-basierten Netzwerken muss eine Implementierung von HRM folgende Bedingungen beachten:

- **IP-Pakete:** Der Transport von Anwendungsdaten geschieht stets auf Basis von IP-Paketen, deren Struktur den offiziellen Definitionen [13] [16] entsprechen und stets gültige Adressen für Quelle und Ziel beinhalten. Dies ermöglicht das Nutzen des Internets als Bindeglied und lässt ebenfalls eine bidirektionale Kommunikation zwischen Knoten aus HRM-basierten Netzen mit Knoten im heutigen Internet zu.
- **IP-Adressierung:** Jeder Knoten im Netzwerk muss über mindestens eine gültige IPv4/IPv6 Adresse identifizierbar sein, insofern er an einer Kommunikation beteiligt sein darf. Diese Bedingung ergibt sich aus Punkt 1. Ist sie nicht erfüllt, können Pakete aus dem Internet nicht dem korrekten Knoten im HRM-Netzwerk zugestellt werden. Umgekehrt gilt die gleiche Einschränkung. Entsprechend Abschnitt 3.4.5 muss zur Erfüllung dieser Bedingung der Netzwerkoperator ein IP-Präfix festlegen, sodass eine einfache Abbildung zwischen HRMIDs und IP-Adressen möglich wird.

Die HRM-Architektur ermöglicht ohne weitere Anpassungen die Erfüllung oben genannter Bedingungen, da ihre internen Signalisierung unabhängig vorhandener Protokolle von Schicht 3 konzipiert sind.

3.9.2 Grenzen bei der Beachtung von Qualitätsanforderungen

Während die Netzabschnitte mit HRM-Unterstützung die Anforderungen der sendenden Anwendung beachten, werden diese beim BE-Routing im Internet ignoriert. Dies kann zu Nachteilen bei der Übertragung von Datenströmen führen, sodass Qualitätsanforderungen nicht eingehalten werden können. Dieser Umstand muss bei der Bewertung der erzielten Routingergebnisse beachtet werden.

3.10 Diskussion der Konzeption

Innerhalb dieses Abschnittes werden die Eigenschaften des Gesamtsystems zusammengefasst und bewertet. Dabei werden insbesondere die im Abschnitt 3.1 aufgestellten Anforderungen an ein neues Routingmanagement mit dem vorgestellten Konzept verglichen und beschrieben, inwiefern die Anforderungen durch die Konzeption erfüllt sind.

3.10.1 Charakterisierung des Gesamtsystems

Analog zu den Abschnitten 2.1.8.4 und 2.3.4.6 kann HRM entsprechend der vorherigen Kriterien charakterisiert werden. Tabelle 3.8 stellt dies im Überblick dar.

Eigenschaft	Hierarchisches Routingmanagement
Einordnung der Signalisierungen im OSI-Modell	ab Schicht 3
Sicherung von Signalisierungen	ja (durch Mechanismen von TCP)
Clustering des Netzwerks	autonom
Hierarchie für Management	autonom
Adressvergabe	autonom
Zielaggregation	ja
Routingdaten	• *RouteReport*: Status lokaler Link • *RouteShare*: Routingtabellen
Art der Verteilung von Routingdaten	• *RouteReport*: hierarchische *Link-State*-Signalisierung • *RouteShare*: Routingtabellen
Ziel der Verteilung von Routingdaten	• *RouteReport*: übergeordneter Koordinator • *RouteShare*: untergeordnete Entitäten
Metrik	Hop-Distanz, aktuell verfügbare Datenrate, aktuell zu erwartende Verzögerung, momentane Auslastung
Routingstrategie	Kontrolldaten (Kontrollebene): • Shortest Path Routing Anwendungsdaten (Datenebene): • 1.Wahl: *Widest Shortest Path Routing* • 2.Wahl: *Shortest Widest Path Routing*
Routenberechnung	*Dijkstra*-Algorithmus
Routingzeitpunkt	proaktives Routing
Lokale Routingtabelle	Intra-AS- & Inter-AS-Routing (in Abhängigkeit von der Netzwerkunterteilung und der Managementhierarchie)
Schleifenvermeidung	ja

Tabelle 3.8: Eigenschaften des hierarchischen Routingmanagements

Wie in Tabelle 3.8 zu sehen, beinhaltet die Konzeption Vorkehrungen gegen Nachrichtenverluste und Permutation der Reihenfolge für die Übertragung von Signalisierungsnachrichten. Zu diesem Zweck werden Mechanismen von TCP eingesetzt.

Eine autonome Arbeitsweise steht bei der Unterteilung des Netzwerks sowie der Verteilung von Adressen und Routingdaten im Vordergrund. HRM verwendet im Gegensatz zu OSPF und BGP für die Verteilung von Routingdaten zwei unterschiedliche Datentypen. Während *RouteReport*-Nachrichten typische Daten einer *Link-State*-Signalisierung enthalten, sind *RouteShare*-Nachrichten vergleichbar mit den Signalisierungsnachrichten von *Distance-Vector*-Protokollen, welche typischerweise Routingtabellen enthalten.

HRM verwendet eine Zweiteilung der Routingstrategie. Innerhalb der Kontrollebene wird durch Auswertung von *AnnounceCoordinator*-Nachrichten stets die kürzeste Route zu einem Knoten gelernt. Somit erfolgt die Weiterleitung von Signalisierungsnachrichten durch *Shortest Path Routing*. Im Gegensatz dazu werden für die Weiterleitung von Anwendungsdaten durch die Datenebene sowohl *Widest Shortest Path* als auch *Shortest Widest Path Routing* eingesetzt. Ähnlich OSPF verwendet HRM den

Dijkstra-Algorithmus zur Bestimmung schleifenloser Routen. Während jeder Routenberechnung werden die Qualitätsanforderungen der sendenden Anwendung beachtet, welche entsprechend der Unterscheidung von Abschnitt 2.2 als *soft* gelten.

3.10.2 Konvergenz und Korrektheit der Platzierung von Managementinstanzen

Bei der Bewertung der Instanziierung der Kontrollebene sind folgende zwei Aspekte interessant:

- Die Strukturierung der Kontrollebene sollte stets in endlicher Zeit[26] abgeschlossen sein.
- Die resultierende Struktur der Managementinstanzen (Koordinatoren) sollte den geforderten Bedingungen genügen. Dazu zählt insbesondere die logische Hop-Distanz zwischen den Instanzen, in Abschnitt 3.10.2.2 werden dazu weitere Details gegeben.

Beide Aspekte werden nachfolgend in Form von allgemeinen Aussagen (Sätze) formuliert. Die Korrektheit dieser Aussagen wird nachfolgend durch direkte Folgerungen hergeleitet, dabei werden aus Einzelaussagen neue Aussagen geschlossen, welche letztlich zur jeweiligen finalen Schlussfolgerung und somit zur Bewertung der ursprünglich getroffenen Aussagen führen.

3.10.2.1 Konvergenz

Aufgrund der Unterschiede zwischen den Phasen 1 und 2 während der Instanziierung der Kontrollebene ist eine grundsätzliche Unterscheidung zwischen dem Basislevel und den höheren Hierarchielevels bei der Betrachtung der Konvergenz sinnvoll.

Annahmen: Die Annahmen des Wahlalgorithmus für das Basislevel (siehe Abschnitt 3.3.2.6) und des Strukturierungsalgorithmus für höhere Hierarchielevels (siehe Abschnitt 3.3.4.9) für Phase 1 bzw. 2 müssen erfüllt sein. Dazu zählt ebenfalls eine zuverlässige Kommunikation innerhalb der Kontrollebene.

Satz 1 (Basislevel): Für das Basislevel wird für ein Netzwerk mit konstanter Topologie in endlicher Zeit eine stabile Struktur von instanziierten L0-Koordinatoren gefunden.

Nachweis für Satz 1:

- **Erkennung der Nachbarschaft und Signalisierung von Knoten-IDs:** Jeder Knoten besitzt eine konstante Knoten-ID, welche er selbst bestimmt hat. Zusätzlich kennt jeder Knoten durch Phase 0 (siehe Abschnitt 3.3.1) in endlicher Zeit seine direkten Nachbarknoten und ihre jeweilige Knoten-ID.
- **Ableitung von lokalen L0-Prioritäten:** Durch den Ablauf von Phase 0 kennt jeder Knoten seine Nachbarn, daraus leitet er seine Konnektivität und somit auch seine L0-Priorität ab. Diese verwendet er für jeden seiner lokalen L0-Clustermanager.
- **Signalisierung von L0-Prioritäten:** Durch den Ablauf von Phase 1 (siehe Abschnitt 3.3.2) kommuniziert jeder L0-Clustermanager seine L0-Priorität an alle anderen L0-Clustermanager seiner jeweiligen Broadcast-Domäne.
- **Bestimmung von Wahlergebnissen:** Die Anzahl von notwendigen Schritten zur Strukturierung des Basislevels der Kontrollebene ist endlich:
    1. Da die Topologie konstant ist, ist die Konnektivität jedes Knotens ebenfalls konstant.

[26] Die Aufstellung einer allgemeingültigen Formel zur Berechnung der erforderlichen Zeit, welche bis zur Ermittlung einer finalen Lösung vergeht, stellt sich als äußerst schwierig dar. Bei einer solchen Betrachtung müssen eine Vielzahl von Freiheitsgraden beachtet werden. Dabei spielen insbesondere die nebenläufigen Signalisierungen im Netzwerk eine Rolle. Zur Reduktion dieser Komplexität bieten sich explizit getroffene Einschränkungen (sogenannte Randbedingungen) an, beispielweise können nur ausgewählte Szenarien mit einer begrenzten Hierarchietiefe und einem festgelegtem Clusterradius betrachtet werden. Des Weiteren kann die Verzögerung bei der Übertragung von Nachrichten vernachlässigt werden. Jedoch wird durch diese Schritte auch die Aussagekraft der resultierenden Formel eingeschränkt.

Folglich sind die L0-Prioritäten aller L0-Clustermanager jedes Knotens konstant. Jeder L0-Clustermanager kommuniziert in endlichen Schritten seine eigene L0-Priorität an eine endliche Menge von L0-Clustermanagern in seiner lokalen Nachbarschaft.

2. Der Wahlalgorithmus leitet durch eine endliche Menge an Vergleichen aus den L0-Prioritäten und Knoten-IDs der Kandidaten einen Wahlsieger ab.
3. Jeder Wahlsieger instanziiert in endlichen Schritten einen L0-Koordinator.

Satz 2 (höhere Hierarchielevels): Für ein höheres Hierarchielevel *n* wird für eine konstante Struktur des jeweils untergeordneten Hierarchielevels *(n - 1)* in endlicher Zeit ebenfalls eine stabile Struktur von instanziierten Koordinatoren gefunden.

Nachweis für Satz 2:

- **Bekanntgabe von Koordinatoren:** Jeder Koordinator gibt kontinuierlich seine Existenz mit Hilfe von *AnnounceCoordinator*-Nachrichten konzentrisch im Netzwerk unter Beachtung des Clusterradius *r* bekannt (siehe Abschnitt 3.3.5). Folglich erreichen jeden Clustermanager des Hierarchielevels *n* die Bekanntgaben aller Koordinatoren des Hierarchielevels *(n - 1)* im jeweiligen Clusterradius *r*.
- **Ableitung von lokalen Prioritäten:** Jeder Knoten besitzt pro Hierarchielevel eine eigene Priorität für Hierarchielevel *n*, diese ist abhängig von der Struktur des jeweils untergeordneten Hierarchielevels *(n - 1)*. Aufgrund der empfangenen *AnnounceCoordinator*-Nachrichten kennt jeder Knoten in endlichen Schritten die Struktur seiner lokalen Nachbarschaft im Radius *r*, sodass daraus ebenfalls in endlichen Schritten die jeweils lokale Priorität ermittelt werden kann.
- **Signalisierung zur Clusterbildung:** Ein Clustermanager auf Hierarchielevel *n* fügt in endlichen Schritten via *RequestClusterMembership*-Signalisierungen alle ihm bekannten untergeordneten Koordinatoren seinem Cluster hinzu.
- **Signalisierung von Prioritäten:** Entsprechend Abschnitt 3.3.4 werden Prioritäten auf höheren Hierarchielevels zwischen einem Clustermanager und seinen untergeordneten Koordinatoren ausgetauscht, sodass jeder Clustermanager stets die Prioritäten seiner untergeordneten Koordinatoren in endlichen Schritten kennt.
- **Bestimmung von Wahlergebnissen:** Die Anzahl von notwendigen Schritten zur Ermittlung einer stabilen Struktur für Hierarchielevel *n* der Kontrollebene ist endlich:
    1. Da das untergeordnete Hierarchielevel *(n - 1)* eine konstante Struktur besitzt, ist die Menge von bekannten Koordinatoren dieses Levels und ihren logischen Distanzen für jeden Knoten ebenfalls konstant. Somit besitzt jeder Clustermanager auf Hierarchielevel *n* ebenfalls eine konstante Priorität, welche er in endlicher Zeit bestimmen kann.
    2. Der Austausch von Prioritäten erfolgt in endlichen Schritten für eine endliche Menge von Kommunikationspartnern.
    3. Der Wahlalgorithmus leitet in endlichen Schritten aus den Prioritäten der beteiligten Wahlteilnehmer und den jeweils zugehörigen Knoten-IDs (diese sind aufgrund von *AnnounceCoordinator*-Nachrichten bekannt) einen Wahlsieger ab.
    4. Jeder Wahlsieger instanziiert in endlichen Schritten einen Koordinator für Hierarchielevel *n*.
    5. Der DCE-Algorithmus verdrängt für jeden Knoten eine endliche Menge von Koordinatoren in endlichen Schritten auf Hierarchielevel *n:*
        - Die Menge von allen übergeordneten Clustermanagern, und somit auch der von übergeordneten Koordinatoren, ist für jeden Koordinator im Radius *r* endlich. Jeder Koordinator auf Hierarchielevel *(n - 1)* bestimmt somit in endlichen Schritten seine übergeordnete Koordinatorinstanz mit der höchsten Priorität.

- Die Signalisierung von *Leave/Return*-Nachrichten eines Koordinators auf Hierarchielevel *(n - 1)* erfolgt für eine endliche Menge von Koordinatoren auf Hierarchielevel *n* im Radius *r*.
- Durch die *Leave/Return*-Nachrichten wird der Wahlalgorithmus für eine endliche Menge von Clustermanagern auf Hierarchielevel *n* neugestartet und in endlicher Zeit (siehe oben) ein neues Wahlergebnis bestimmt. In Abhängigkeit von dem neuen Ergebnis wird eine endliche Anzahl von Koordinatorinstanzen auf Hierarchielevel *n* entfernt.

Schlussfolgerungen: Bei der Instanziierung der Kontrollebene ist das Ergebnis für ein höheres Level *n* ausschließlich von der Struktur von Level *(n - 1)* abhängig, sodass eine Abhängigkeit von der Struktur von Level *(n + 1)* **nicht** existiert. Es existiert somit keine zyklische Datenabhängigkeit zwischen den Hierarchielevels. Folglich sind bei einer konstanten Netzwerktopologie sowohl die Eingaben als auch die darauf ausgeführten Berechnungsschritte für jedes Hierarchielevel endlich. Folglich steht ebenfalls fest, dass der allgemeine Instanziierungsvorgang der Kontrollebene für eine konstante Netzwerktopologie in endlicher Zeit terminiert. Dabei konvergiert die Strukturierung mit einer finalen Lösung, sodass nachfolgend die Verteilung der Koordinatoren auf allen Hierarchielevels stabil verbleibt, bis wiederum neue Topologieänderungen eintreten. Die ursprünglich getroffenen Aussagen sind korrekt.

3.10.2.2 Korrektheit

Annahmen: Für die Korrektheit gelten die Annahmen der Konvergenzbetrachtung (siehe vorheriger Abschnitt). Dazu zählt insbesondere eine konsistente Kommunikation innerhalb der Kontrollebene.

Satz 1: Für das Basislevel wird für ein Netzwerk mit konstanter Topologie stets die korrekte Struktur als finale Lösung ermittelt. In dieser ist ein L0-Koordinator immer auf dem Knoten mit der für die jeweilige Broadcast-Domäne höchsten Konnektivität instanziiert. Gibt es mehrere Knoten mit gleicher Konnektivität werden zusätzlich die Knoten-IDs einbezogen und die Koordinatorinstanz wird auf dem Knoten mit der höchsten ID erstellt.

Nachweis:

- **Ableitung von L0-Prioritäten:** Aus Phase 0 kennt jeder Knoten seine lokale Nachbarschaft, daraus leitet er seine eigene Konnektivität und daraus wiederum seine L0-Priorität ab. Diese steigt proportional zur Konnektivität und wird von den jeweils lokalen Clustermanagern für die Wahlvorgänge verwendet.
- **Totale Ordnung:** Zur Bestimmung eines eindeutigen Wahlergebnisses muss stets eine totale Ordnung unter den Kandidaten hergestellt werden können. Dazu verwendet der Wahlalgorithmus aus Abschnitt 3.3.2 die L0-Prioritäten als primäres Kriterium. Sollte es mehrere Kandidaten mit gleicher Priorität geben, werden zusätzlich die jeweiligen Knoten-IDs als sekundäres Kriterium ausgewertet und es gewinnt der Clustermanager, dessen zugehöriger Knoten die höchste ID besitzt.
- **Resultierende Struktur:** Die resultierende Struktur ist korrekt:
    1. Der Wahlalgorithmus wählt entsprechend der totalen Ordnung pro Broadcast-Domäne stets den L0-Clustermanager mit der höchsten L0-Priorität respektive Knoten-ID als Wahlsieger aus.
    2. Jeder Wahlsieger instanziiert einen L0-Koordinator, sodass ein L0-Koordinator für eine Broadcast-Domäne immer auf dem Knoten mit der höchsten Priorität, und somit auch der höchsten Konnektivität, erstellt wird. Gibt es mehrere Knoten mit den gleichen Werten wird anhand der Knoten-IDs entschieden.

Satz 2: Für ein höheres Hierarchielevel *n* wird stets die korrekte Struktur als finale Lösung gefunden, insofern das untergeordnete Hierarchielevel (*n* - 1) eine konstante, korrekte Struktur besitzt. Dabei gilt die Struktur eines höheren Hierarchielevels *n* als korrekt, wenn:

- Ein Koordinator wird nur auf Knoten erstellt, auf denen bereits ein Koordinator des untergeordneten Hierarchielevels (n - 1) existiert.
- Für jeden Koordinator gilt, dass er in Bezug auf die aktiven Wahlmitgliedschaften die höchste Priorität und, bei identischen Prioritäten, auch die höchste Knoten-ID innerhalb des Radius *r* zugeordnet hat.

$$r + 1 \leq min\_d_{i,j} \leq 2 * r + 1$$

Abbildung 3.51: Minimale Distanz zwischen zwei benachbarten Koordinatoren

- Zusätzlich muss für zwei benachbarte Koordinatoren *Ci* und *Cj* auf dem Hierarchielevel *n* stets die Ungleichung aus Abbildung 3.51 gelten. Sie beschreibt die erlaubten Werte für die minimale Distanz $min\_d_{i,j}$ auf Hierarchielevel *(n - 1)* zwischen den Koordinatoren. Dabei wird die jeweilige Distanz zwischen zwei Koordinatoren aus der Anzahl der zu passierenden Clustern des jeweils untergeordneten Hierarchielevels *(n -1)* gebildet. Folglich kann sie auch als logische Hop-Distanz zwischen den Koordinatoren bezeichnet werden.

Nachweis:

- **Ableitung von Priorität:** Die Priorität eines Knotens für ein höheres Hierarchielevel *n* ist abhängig von der Struktur des untergeordneten Hierarchielevels *(n - 1)*. Je mehr und näher sich untergeordnete Koordinatoren im Umkreis *r* eines Knotens befinden, desto höher ist seine Priorität für Hierarchielevel *n*. Folglich ist beispielsweise Level 1 abhängig von der Platzierung der Koordinatoren des Basislevels.
- **Totale Ordnung:** Analog zum Basislevel muss auch für die Bestimmung eines eindeutigen Wahlergebnisses für höhere Hierarchielevels eine totale Ordnung unter den Kandidaten hergestellt werden können. Dazu werden (siehe Abschnitt 3.3.4) ausschließlich die Prioritäten sowie die jeweiligen Knoten-IDs der Kandidaten verwendet. Sollte es mehrere Kandidaten mit gleicher Priorität geben, werden zusätzlich die jeweiligen Knoten-IDs als sekundäres Kriterium ausgewertet und es gewinnt der Kandidat, dessen zugehöriger Knoten die höchste ID besitzt.
- **Resultierende Struktur:** Die resultierende Struktur ist korrekt, da:
    1. Ein Clustermanager eines höheren Hierarchielevels *n* wird entsprechend der Konzeption nur auf Knoten erstellt, welche bereits eine Koordinatorinstanz des untergeordneten Hierarchielevels *(n - 1)* besitzen. Da nur Clustermanager eine Koordinatorinstanz erstellen können, werden somit die Koordinatorinstanzen für Hierarchielevel *n* auch nur auf Knoten erstellt, welche bereits eine lokale Koordinatorinstanz des untergeordneten Hierarchielevels *(n - 1)* besitzen.
    2. Der Wahlalgorithmus wählt einen Clustermanager entsprechend der hergestellten totalen Ordnung als Sieger aus, sodass dieser die höchste Priorität respektive Konten-ID innerhalb seines Radius *r* besitzt.
    3. Jeder Wahlsieger instanziiert einen Koordinator auf Hierarchielevel *n*.
    4. Mit Hilfe des DCE-Algorithmus verdrängt jede Koordinatorinstanz durch *Leave/Return*-Nachrichten alle umliegenden Instanzen mit niedriger Priorität im Umkreis *r*. Daraus ergibt sich ein minimaler Abstand (die logische Hop-Distanz) von *(r + 1)* zwischen allen resultierenden Koordinatoren eines höheren Hierarchielevels *n*.
    5. Da *AnnounceCoordinator*-Nachrichten zur Bekanntgabe eines Koordinators von Hierarchielevel *(n - 1)* nur übergeordneten Clustermanager im Radius *r* erreichen, tauscht

ein Koordinator ausschließlich mit übergeordneten Clustermanagern im Radius *r* Nachrichten aus und wird (je nach Aktivierung der jeweiligen Wahlmitgliedschaft) in deren Wahlvorgängen beachtet. Unter diesen Clustermanagern wählt ein Koordinator stets denjenigen mit aktiver Koordinatorinstanz aus, der die höchste Priorität besitzt. Bei diesem hat der Koordinator die zugehörige Wahlmitgliedschaft mit Hilfe des DCE-Algorithmus aktiviert. Folglich besitzt jeder Koordinator auf Hierarchielevel *(n - 1)* einen übergeordneten Koordinator auf Hierarchielevel *n* innerhalb des Radius *r*. Für zwei benachbarte Koordinatoren auf Hierarchielevel *(n - 1)* mit einer logischen Distanz von 1 ergibt sich dadurch eine maximale, logische Hop-Distanz von *2 * r + 1* für die jeweils übergeordneten Koordinatoren auf Hierarchielevel *n*.

Schlussfolgerungen: Da alle beschriebenen Einzelschritte der Strukturierung der Kontrollebene unabhängig von der Eingabe deterministisch sind, ist das resultierende Ergebnis ebenfalls deterministisch. Für das Basislevel wurde nachgewiesen, dass L0-Koordinatoren nur auf Knoten mit der für die jeweilige Broadcast-Domäne höchsten Konnektivität instanziiert werden. Sollten mehrere die gleichen Werte besitzen, werden zusätzlich die Knoten-IDs einbezogen und es gewinnt der Knoten mit der höchsten ID. Für höhere Hierarchielevels wurde nachgewiesen, dass die finale Lösung für die Strukturierung der Kontrollebene stets die Bedingungen aus Satz 2 erfüllt. Somit steht fest, dass die ursprünglich getroffenen Aussagen korrekt sind.

3.10.3 Vollständigkeit der Adresszuweisung

Interessant ist bei der Adressvergabe die Frage, ob durch die Kontrollebene jeder Netzwerkschnittstelle im Netzwerk eine Adresse zugewiesen wird. Zur Beantwortung dieser Frage muss die Strukturierung der Kontrollebene analysiert werden.

Satz: Jede Netzwerkschnittstelle erhält durch die Kontrollebene eine HRMID zugewiesen.

Nachweis:

- **Broadcast-Domänen:** Jede Netzwerkschnittstelle eines Knotens gehört einer Broadcast-Domäne an.
- **Ein L0-Clustermanager je Broadcast-Domäne:** Für jede Broadcast-Domäne besitzt ein Knoten einen separaten L0-Clustermanager.
- **Eingliederung aller L0-Clustermanager in die Kontrollebene:** Jeder L0-Clustermanager ist Mitglied der Kontrollebene
- **Resultierende Adressverteilung:**
    1. Die Adressvergabe startet am TOP-Koordinator und endet an den Blättern der Hierarchie. Diese werden durch die L0-Clustermanager repräsentiert. Dadurch erhält jeder Knoten für jeden existierenden L0-Clustermanager – und somit auch für jede Netzwerkschnittstelle – eine eindeutige HRMID zugewiesen.
    2. Wird bei einer dynamischen Topologie ein Knoten dem Netzwerk hinzugefügt, erstellt er automatisch lokale L0-Clustermanager, welche ebenfalls automatisch in die Kontrollebene eingegliedert werden. Folglich erhält der neue Knoten separate, eindeutige HRMIDs für alle Netzwerkschnittstellen zugewiesen.

Schlussfolgerungen: Da jeder Knoten automatisch L0-Clustermanger erstellt, welche automatisch der Kontrollebene hinzugefügt werden, erhält er über diese für jede seiner Netzwerkschnittstellen eine separate HRMID zugewiesen. Die ursprünglich getroffene Aussage ist korrekt.

3.10.4 Implikationen von Entscheidungen einzelner Kernkomponenten

Die HRM-Architektur besteht aus verschiedenen Kernkomponenten, wobei die Entscheidungen einer Komponente das Ergebnis von anderen beeinflussen kann. Dabei sollten Zyklen in den Abhängigkeiten vermieden werden, um die Konvergenz von Abläufen in endlicher Zeit zu gewährleisten.

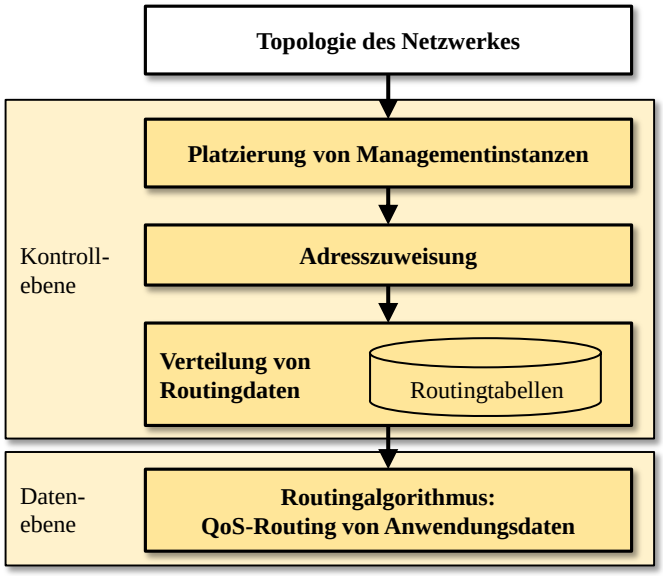

Abbildung 3.52: Abhängigkeiten der Kernkomponenten der Architektur

In Abbildung 3.52 ist zu erkennen, dass die Kernkomponenten keine zyklische Abhängigkeit aufweisen. Stattdessen ergibt sich eine Kette von Abhängigkeiten zwischen den Komponenten der Architektur:

- **Platzierung von Managementinstanzen**: Aus der Topologie des Netzwerks ergibt sich die Konnektivität von einzelnen Knoten, wodurch die jeweilige L0-Priotität abhängt. In Abhängigkeit von der resultierenden Prioritätsverteilung werden durch die Abläufe in der Kontrollebene die Koordinatoren im Netzwerk auf unterschiedlichen Hierarchielevels verteilt.
- **Adresszuweisung**: Die Platzierung der Managementinstanzen sowie die Zuordnung der Koordinatoren zwischen verschiedenen Hierarchielevels beeinflusst die Struktur der Hierarchie. Da die Adressen wiederum durch die einzelnen Entitäten der Hierarchie im Netzwerk verteilt werden, beeinflusst die Struktur der Hierarchie auch die Verteilung von Adressen im Netzwerk.
- **Verteilung von Routingdaten**: Die einzelnen Entitäten kommunizieren in Richtung der Wurzel und der Blätter der Hierarchie ausgewählte Routingdaten. Diese beinhalten aggregierte Routen, welche Pfade zu Knotengruppen beschreiben. Diese Gruppierungen sind abhängig von der Adresszuweisung und der daraus resultierenden Verteilung von Adressen, da immer nur ähnliche Adresse (entsprechend der Clusterunterteilung des Netzwerks) zusammengefasst werden können. Aus den signalisierten Routen bildet jeder Knoten seine lokale Routingtabelle, welche er auf Basis der kontinuierlichen empfangenen *RouteReport*- und *RouteShare*-Nachrichten stetig aktuell hält. Dadurch beinhalten die Tabellen eine stets genaue Übersicht über bekannte Routen und ihre aktuell verfügbaren Kapazitäten.
- **Routingalgorithmus**: Da der Algorithmus die knotenlokale Routingtabelle als Eingabe verwendet, ist sein Ergebnis direkt abhängig von der Genauigkeit und der Aktualität der darin enthaltenen Daten. Sollten durch die verwendete Ziel- und Topologieaggregation einzelne Details über die physikalisch vorhandene Topologie reduziert worden sein, kann dadurch die Routingentscheidung beeinflusst werden.

Im Gegensatz zu den zuvor aufgeführten Komponenten ist das Ergebnis eines Routingmanagers nicht von den anderen Komponenten abhängig. Er führt stets die gleichen Schritte aus und das Resultat ist die HRMID des jeweils nächsten Knotens in Richtung des jeweiligen Paketziels.

3.10.5 Stabilität des Routings

Für stabile Routingentscheidungen sind möglichst gleich bleibende Routingdaten notwendig. Einträge von Routingtabellen sollten durch aufeinanderfolgende Aktualisierungsnachrichten möglichst nicht zyklisch gelöscht und wieder neu gespeichert werden. Um dies zu verhindern, müssen sowohl die Konnektivität als auch die zugewiesenen Adressen im Netzwerk möglichst konstant gehalten werden. Die Konnektivität kann sich jedoch im Netz jederzeit ändern und eine Umstrukturierung der Kontrollebene verursachen. Dies kann ebenfalls zum kurzzeitigen Ausbleiben von Aktualisierungsnachrichten für Routen führen, welche nicht von den Topologieänderungen betroffen sind. HRM leistet in dem Fall eine Zwischenpufferung der bekannten Daten, sodass diese erst nach Ablauf ihrer Gültigkeit endgültig gelöscht werden. Des Weiteren unterstützt die in Abschnitt 3.4.4 beschriebene Signalisierung eine stabile Adressvergabe bei Veränderungen im Netzwerk, sodass dadurch die Stabilität von Routingdaten bei dynamischer Topologie weitestgehend gewährleistet wird.

3.10.6 Verzögerung von Routingdaten

Eine korrekte Routingentscheidung hängt insbesondere von den Eingabedaten ab. Dazu zählen die lokal bekannten Routen und ihre QoS-Eigenschaften. Es ist dabei wichtig, dass diese Daten möglichst dem aktuellen Zustand des Netzwerks und der vorhandenen Linkkapazitäten entsprechen. Im schlechtesten Fall sind bei der Signalisierung der für eine Routingentscheidung notwendigen Routingdaten alle Hierarchielevels beteiligt. In diesem Fall beginnt die Signalisierung an einem L0-Clustermanager, setzt sich über seinen L0-Koordinator bis zum TOP-Koordinator fort, und endet in umgekehrter Richtung wiederum an einem anderen L0-Clustermanager. Die resultierende Verzögerung der vollständigen Signalisierung ist dabei von der Zeit T_{min} zwischen zwei aufeinanderfolgenden *RouteReport*- bzw. *RouteShare*-Nachrichten abhängig. Nähere Details sind dazu in Abschnitt 3.5.4.2 zu finden. Zusätzlich ergeben sich Verzögerungen durch die notwendigen Netzwerkübertragungen. Im ungünstigsten Fall befinden sich die pro Hierarchielevel beteiligten Instanzen der Kontrollebene auf unterschiedlichen Knoten, sodass für jede einzelne Signalisierung zusätzliche Verzögerungen auftreten.

$$t_{max\_delay} = H * (T_{min\_RR} + T_{min\_RS}) + t_{sum\_delay\_E2E}$$

Formel 3.9: Maximale Verzögerung der Verteilung von Routingdaten für den längsten Signalisierungsweg

Formel 3.9 beschreibt die resultierende maximale Verzögerung t_{max_delay} für eine Ende-zu-Ende Aktualisierung von Routingdaten. Dabei werden folgende Eingaben verwendet:

- **H:** Tiefe der verwendeten Hierarchie der Kontrollebene
- $\mathbf{T_{min\_RR}}$**:** minimale Zeit zwischen zwei aufeinanderfolgender *RouteReport*-Nachrichten
- $\mathbf{T_{min\_RS}}$**:** minimale Zeit zwischen zwei aufeinanderfolgender *RouteShare*-Nachrichten
- $\mathbf{t_{sum\_delay\_E2E}}$**:** Summe aller Verzögerungen während der Übertragung von Signalisierungen

Die Zeiten T_{min_RR} und T_{min_RS} beschreiben dabei den Fall, dass der jeweilige Signalisierungszyklus bereits kurz zuvor ausgeführt wurde und so die vollständige Wartezeit zwischen zwei aufeinanderfolgenden Signalisierungen erneut abgewartet werden muss (dies gilt sowohl für Teil- als auch Vollaktualisierungen). Nimmt man beispielsweise beide Zeiten mit einem Wert von jeweils 1 Sekunde an und schätzt die auftretenden Netzwerkverzögerungen mit 60 ms ein (6 Übertragungen mit jeweils 10 ms Verzögerung), erhält man für eine Hierarchietiefe von 3 eine maximale Verzögerung von 6,06 Sekunden. Das erscheint akzeptabel, da dieser Fall nur eintritt, wenn die Wartezeit bei allen Signalisierungsschritten zutrifft. Für einen Wert von 5 Sekunden für die Zeiten T_{min_RR} und T_{min_RS} ergibt sich dabei bereits eine

maximale Verzögerung von 30,06 Sekunden. Dieser Zusammenhang muss bei der Wahl der Wartezeiten für *RouteReport/RouteShare*-Nachrichten beachtet werden[27].

3.10.7 Rerouting bei Topologieänderungen

Sollten ein Link oder Knoten im Netz ausfallen, werden die betroffenen Routen entsprechend der in Abschnitt 3.5.4.3 erläuterten maximalen Zeit innerhalb der Routingtabellen aller Knoten automatisch gelöscht. Bei Verwendung des *DiffServ*-Modells führt dies zu einem automatischen Rerouting entlang alternativer Routen.

Sofern das *IntServ*-Modell und ein damit verbundenes Reservierungsprotokoll eingesetzt werden, muss die bisherige Route auf Basis einer Alternativroute teilweise oder vollständig ersetzt werden. Dabei ist es Aufgabe des verwendeten Protokolls die notwendige Signalisierung entweder automatisch auf dem Fehler-erkennenden Router oder am sendenden Knoten auszulösen. Die Datenebene beantwortet die dadurch verursachten Routinganfragen, sodass die notwendigen Reservierungen neu aufgebaut und die Übertragung fortgesetzt werden kann.

3.10.8 Typische Szenarien und Grenzen der Anwendung von HRM

Wie jedes andere System hat auch HRM bevorzugte Einsatzgebiete und Szenarien, in denen es seine Vorteile nur schwer oder gar nicht erbringen kann. Im Allgemeinen ist HRM insbesondere in Netzwerken sinnvoll, in denen das Management des Netzwerks möglichst automatisiert erfolgen soll. Dies kann sowohl ökonomisch als auch anwendungsspezifisch motiviert sein. Im ersten Fall kann das ein Internetprovider oder eine ähnlich große Firma mit entsprechend dimensionierter Netzwerkinfrastruktur sein, welche mit möglichst geringen Kosten zu verwalten sein muss. Der zweite Fall kann beispielsweise bei einem großflächigen Rettungsszenario eintreten, bei dem die einzelnen Gruppen von Einsatzkräften über ein Netzwerk miteinander kommunizieren müssen. Das Netzwerk kann dabei beispielsweise aus einfachen WLAN-Routern gebildet werden, zwischen denen möglichst unmittelbar nach Einsatzstart und ohne Konfigurationshürden geroutet werden muss, sodass eine Koordination der Rettungskräfte ermöglicht wird. Sowohl im Firmen- als auch im Rettungsnetzwerk kann QoS eine zusätzliche zentrale Rolle spielen. Dabei erscheint der Einsatz von HRM besonders für Netzwerke mit vielen redundanten Links sinnvoll, sodass das durch HRM erbrachte Routing die Auswahl von Alternativwegen in Abhängigkeit von der aktuellen Lastverteilung im Netzwerk ausführen kann. Neben den typischen audiovisuellen Übertragungen beim Einsatz von Videokonferenzsystemen oder bei Echtzeitüberwachung entfernter Produktionsabläufe können beispielsweise auch zeitkritische Synchronisationen von Datenbanken (z.B. Kartendaten im Rettungsszenario) davon profitieren.

Da sich die Kontrollebene von HRM dynamisch an topologische Veränderungen anpasst, ist ein Einsatz für hochdynamische Netzwerke eher mit Einschränkungen verbunden. Sollten sich die Strukturen sehr oft ändern, kann dies bei großen Netzwerken zu hohem Signalisierungsaufkommen der Kontrollebene führen und sich dadurch das Kosten-Nutzen-Verhältnis von HRM signifikant verschlechtern. Um dem entgegen zu wirken, können manuell zusätzliche Knotengewichte vergeben werden. Sie fließen als zusätzliche Faktoren in die Berechnung von Prioritäten auf allen Hierarchielevels ein und schränken die autonome Verteilung der HRM-Managementinstanzen gezielt ein.

3.10.9 Bewertung des Gesamtsystems

Zum Ende der Diskussion der Konzeption ist ein Vergleich des Gesamtsystems mit den zuvor in Abschnitt 3.1 aufgestellten Anforderungen interessant. Dabei wird deutlich, dass die geforderten Kernfunktionen durch das Konzept umgesetzt werden:

[27] In dieser Arbeit wird daher eine Wartezeit von 1 Sekunde favorisiert und für die Messungen in Abschnitt 6.3.1 verwendet.

- **Adresszuweisung:** Die Koordinatoren der Kontrollebene weisen jeder Netzwerkschnittstelle eines Knotens eine HRMID zu. Diese HRMIDs sind global eindeutig und können zur Beschreibung einer Route sowie eines Zielknotens verwendet werden.
- **Verteilung von Routingdaten:** Über die Koordinatoren der Kontrollebene werden Routingdaten zwischen den Knoten verteilt, sodass auf jedem Knoten eine Routingtabelle existiert. Sie dient für den Routingalgorithmus als Eingabe und beinhaltet ausreichend Daten für eine Übertragung zu jedem bekannten Ziel im Netzwerk.
- **Routingalgorithmus:** Der Routingalgorithmus wird pro Knoten angewendet, er verwendet sowohl die Anforderungen der jeweiligen Anwendung als auch die lokal vorliegende Routingtabelle als Eingabe. Diese enthält eine Beschreibung der aktuellen Eigenschaften von bekannten Routen, sodass auf Basis dieser Daten ein erfolgreiches QoS-Routing für Anwendungsdaten umgesetzt werden kann.

Die zusätzlich geforderten Eigenschaften sind im Konzept wie folgt beachtet:

- **Autonomität:** Alle beschriebenen Abläufe erfolgen vollkommen automatisch und benötigen keine manuellen Eingaben, dennoch sind sie durch einen Netzwerkadministrator beeinflussbar, indem dieser die verwendeten Parameter entsprechend anpasst. Er kann beispielsweise den globalen Konfigurationsparameter Clusterradius festlegen und somit die Größe resultierender Cluster beeinflussen. Dieses Thema wird detailliert in Abschnitt 6.2.1.1 analysiert.
- **Kompatibilität:** Die HRM-Signalisierung verwendet keine IP-Adressen. Entsprechend Abschnitt 3.6 benötigt sie ausschließlich ein Protokoll der Sicherungsschicht zur Nachrichtenübermittlung zwischen den Netzwerkknoten. Dadurch wird eine hohe Kompatibilität erreicht: HRM kann mit IP kombiniert werden, ebenfalls kann es für FoG-basierte Netzwerke eingesetzt werden. Nähere Details dazu werden im Implementierungskapitel 4 gegeben.
- **Skalierbarkeit:** Für die Signalisierungen wird das Netzwerk in Cluster mit begrenztem Durchmesser[28] unterteilt. Zur Verwaltung dieser Netzwerkabschnitte wird eine Hierarchie eingesetzt, welche die Signalisierungen und Berechnungen über die Netzwerkknoten verteilt. Des Weiteren erfolgt der Austausch von Routen zwischen den Managementinstanzen der Hierarchie in aggregierter Form, sodass dadurch die Menge der verursachten Signalisierungsdaten reduziert wird. Analog dazu werden die verteilten Routen auf jedem Knoten ebenfalls in aggregierter Form innerhalb der Routingtabellen gespeichert.
- **Modularität:** Die einzelnen Prozesse der Kontrollebene laufen eigenständig ab. Es gibt chronologische Abhängigkeiten – Details sind dazu in Abschnitt 3.10.4 zu finden. Jedoch werden zyklische Abhängigkeiten zwischen den verschiedenen Komponenten von HRM vermieden.

Aufgrund der oben aufgeführten Punkte erfüllt HRM alle Anforderungen aus Abschnitt 3.1. Die getroffenen Aussagen werden zusätzlich in Kapitel 6 durch empirische Erkenntnisse anhand ausgewählter, implementierter Netzwerkszenarien unterstützt.

3.11 Vergleich mit bekannten Ansätzen

In Kapitel 2 fällt in den Beschreibungen für heutige und zukünftige Netzwerke insbesondere ein Mangel an autonomer Arbeitsweise auf. HRM schließt diese Lücke durch seine selbststrukturierende Kontrollebene, welche die Basis für das QoS-Routing der Datenebene bildet. Ein detaillierter Vergleich von HRM mit bisherigen Ansätzen hinsichtlich aller erdenklichen Kriterien würde den Rahmen dieser Arbeit

[28] Als Durchmesser wird das Maximum der kürzesten Routen zwischen zwei beliebigen Knoten des Clusters verstanden.

sprengen. Stattdessen werden im Folgenden ausgewählte Kriterien verwendet und auf folgende Alternativansätze angewandt:

- **OSPF und BGP:** Als typische Ansätze heutiger Netzwerke erscheint ein Vergleich mit HRM sinnvoll. Dabei sollte HRM keine signifikanten Nachteile aufweisen, sodass nichts gegen einen Einsatz in heutigen Netzwerken spricht.
- **PNNI und HSR:** Beide wenden eine Managementhierarchie an, zu denen im Vergleich zu anderen Alternativansätzen HRM die größte Ähnlichkeit aufweist.
- **Selective Probing:** Die in HRM verwendete Routingstrategie ähnelt diesem Ansatz. Des Weiteren stellen *Probe*-Nachrichten eine sinnvolle Erweiterung für HRM zur Ermittlung von passenden Routen dar.
- **Simulierter Routingdienst für FoG:** FoG unterstützt modulares Routing und eignet sich dadurch zur Kombination von verschiedenen zukünftigen Routingprotokollen. Des Weiteren existierte bereits zum Zeitpunkt dieser Arbeit ein implementierter Routingdienst, welcher ohne ein Signalisierungsprotokoll arbeitet und ausschließlich auf Basis von direkten Funktionsaufrufen Daten zwischen den Knoten verteilt. Er wird deshalb als sogenannter simulierter Routingdienstbezeichnet und bietet ebenfalls eine Hierarchieunterstützung. Folglich ist es sinnvoll, HRM mit diesem simulierten Routingdienst für FoG zu vergleichen.

Tabelle 3.9 beinhaltet eine Zusammenfassung der heutigen Routingprotokolle OSPF sowie BGP aus Abschnitt 2.1.8.4. Zusätzlich werden die drei Alternativlösungen PNNI, HSR sowie *Selective Probing* charakterisiert. Des Weiteren werden die in Abschnitt 2.3.4 erläuterten Details zum simulierten Routingdienst von FoG im Überblick dargestellt. Innerhalb der Tabelle wird an einigen Stellen keine Angabe („kA“) gemacht, da die in der Literatur dargestellten Details nicht für eine Einschätzung dieser Eigenschaften genügen. Die Signalisierungen von OSPF und seinen QoS-spezifischen Erweiterungen aus Abschnitt 2.2.6.1 weisen eine sehr hohe Ähnlichkeit auf, die jeweils verwendeten Annahmen unterscheiden sich ebenfalls nicht grundlegend voneinander, daher sind die verfügbaren Erweiterungen für OSPF innerhalb der Tabelle nicht separat aufgeführt. Ähnliches gilt für BGP und seiner in Abschnitt 2.2.6.2 vorgestellten Erweiterung namens QBBP. Ebenfalls sind Overlay-Architekturen, bspw. OverQoS [110] oder SpoVNet [111] [112], nicht separat aufgeführt. Ihre Signalisierungen verwenden das Routing der zugrundeliegenden Protokolle als Basis und sind somit von dessen Performanz abhängig. Dies trifft ebenfalls für die Weiterleitung von Paketen mit Hilfe von festgelegten *Traffic Trunks* unter Verwendung von MPLS-TE [113] sowie beim Einsatz von RSVP-TE [114] zu.

Eigenschaft	**OSPF**	**BGP (inter-AS)**	**PNNI**	**HSR**	**Selective Probing [85]**	**Simulierter Routingdienst für FoG**	**HRM**
Einordnung der Signalisierungen im OSI-Modell	Schicht 4	Schicht 5/6/7	Schicht 2	kA	kA	kA	ab Schicht 3
Sicherung von Signalisierungen	ja	ja	ja	kA	kA	kA	ja
Clusterbildung	ja (manuelle Eingabe)	ja (manuelle Eingabe)	ja (manuelle Eingabe)	kA	kA	ja (manuelle Eingabe)	ja (autonome Erstellung)
Hierarchiebildung	ja (manuelle Eingabe)	ja (manuelle Eingabe)	ja (manuelle Eingabe)	kA	kA	ja (manuelle Eingabe)	ja (autonome Erstellung)
Adresszuweisung	nein	nein	nein (erfolgt manuell für ATM-Switches)	ja	nein	kA	ja
Zielaggregation	ja	ja	ja	ja	kA	kA	ja
Verteilung von Routingdaten	*Link-States*	Routingtabellen mit AS-Pfaden	*Link-States*	kA	kA	kA	hierarchische *Link-States*, Routingtabellen
Metrik	Hop-Distanz	Länge des AS-Pfads, Hop-Distanz zum nächsten Router, Priorität	Hop-Distanz, QoS-Eigenschaften, Auslastung, Netzwerkrichtlinie	kA	Hop-Distanz, QoS-Eigenschaften, Kosten nach Netzwerkrichtlinie	Hop-Distanz	Hop-Distanz, QoS-Eigenschaften, Auslastung
Routingstrategie	*Shortest Path*	*Shortest Path*, Beachtung von Netzwerkrichtlinien	*Shortest Path*	Routing gleich der Hierarchie	1.Iteration: *Shortest Path*, 2.Iteration: First Match	*Shortest Path*	Kontrolldaten: *Shortest Path*, Anwendungsdaten: WSPF / SWPF
Beachtung von Qualitätsanf. beim Routing	nein	nein	ja	nein	ja	nein	ja
Routenberechnung	*Dijkstra*-Algo.	regelbasiert	*Dijkstra*-Algo.	feste Routen	*Probe*-basierte Routenfindung	*Dijkstra*-Algo.	*Dijkstra*-Algo.
Routingzeitpunkt	proaktiv	proaktiv	reaktiv	reaktiv	reaktiv	reaktiv	proaktiv
Erreichbare Routingziele	alle Knoten des AS	alle AS	alle Knoten	alle Knoten	alle Knoten	alle Knoten mit Namen	alle Knoten
Möglichkeit zur Vermeidung von Routingschleifen	ja	ja	ja	ja	ja	ja	ja

Tabelle 3.9: Vergleich von HRM mit Alternativlösungen für heutige und zukünftige Netzwerke

Die in Tabelle 3.9 fett dargestellten Eigenschaften betreffen die Kernkomponenten von HRM, sie werden im Folgenden für eine genauere Abgrenzung des Konzeptes gegenüber bisherigen Ansätzen detaillierter beschrieben. Dabei werden an geeigneten Stellen zusätzliche Alternativansätze aufgeführt, welche Ähnlichkeiten zu HRM besitzen.

3.11.1 Clusterbildung

Im Gegensatz zu bekannten divisiven Algorithmen zur Clusterunterteilung [115] [116] arbeitet HRM nach der agglomerativen Verfahrensweise [104]. Dabei wird die Hierarchie von den Blättern in Richtung der Spitze durch Vergrößern von Clustern aufgebaut. Für höhere Hierarchielevel ist die Clusterbildung von HRM vergleichbar mit dem *Zone Routing Protocol* (ZRP) [90] und seinen verwendeten Zonen. Dabei startet jeder Knoten als Zentrum seiner eigenen Zone. Jeder Knoten fügt zu seiner Zone wiederum Knoten der jeweils lokalen Nachbarschaft hinzu. Es werden nur Knoten beachtet, deren logische Hop-Distanz kleiner als ein definierter Zonenradius r_{zone} ist. Dies gleicht dem bei HRM eingesetzten Radius *r*. Während bei ZRP alle Mitglieder einer Zone zu beliebig vielen anderen Clustern gehören dürfen, wird bei HRM zusätzlich der DCE-Algorithmus eingesetzt. Durch seine Signalisierungen verdrängen sich Koordinatorinstanzen gegenseitig, sodass der Abstand zwischen ihnen immer größer als der Radius *r* ist. Des Weiteren gehört ein Koordinator bei HRM ausschließlich einem übergeordneten Cluster mit aktiver Koordinatorinstanz an. Die eigenständige Wahl eines übergeordneten Koordinators bei HRM ähnelt dabei dem Registrierungsprozess von *Cluster Based Routing* (CBR) [117]. Darin wird die Signalstärke als Kriterium der Zuordnung zu einem *Cluster Head* (Koordinator) verwendet. Nach erfolgter Entscheidung für einen Kandidaten wird dieser per Registrierungsanfrage darüber informiert. Im Gegensatz zu HRM verwendet CBR die *Cluster Heads* auch für die Weiterleitung von Anwendungsdaten und lässt somit eine Unterscheidung zwischen Kontroll- und Datenebene vermissen. Dadurch werden bei CBR die Routingentscheidungen durch die Struktur der *Cluster Heads* vorgegeben.

Vergleicht man HRM mit den heutigen Protokollen OSPF oder BGP, fällt insbesondere die verwendete Netzwerkunterteilung auf. Beide Routingprotokolle beruhen auf einer vorgegebenen Struktur und unterscheiden sich dadurch erheblich von HRM. PNNI und seine als *Peer Groups* bezeichneten Cluster beinhalten eine ähnliche Annahme. Für die eindeutige Clusterunterscheidung kommen sogenannte *Peer Group Identifier* (PGI) und *Level Indicators* (LI) auf unterschiedlichen Hierarchielevels zum Einsatz. Beide Werte sind in der ATM-Adresse eines Switches enthalten und bestimmen die Clusterzugehörigkeit eines Switches. Diese Form der Adressierung ähnelt der Angabe von Entität-ID sowie Hierarchielevel für eine Entität der Kontrollebene von HRM. Im Gegensatz zu dem automatisierten Ansatz von HRM sind bei PNNI die ATM-Adressen vorgegeben.

Ein Vergleich mit HSR, als Vertreter zukünftiger Ansätze, ist nur eingeschränkt möglich. Die Literatur spezifiziert nicht detailliert, wie ein Netzwerk in Cluster unterteilt wird. Stattdessen werden abstrakt beschriebene Möglichkeiten erwähnt. Dabei wird zwischen physikalischer und logischer Partitionierung unterschieden. Erstere wird in Verbindung zur geographischen Lokalisation gebracht, während die logische Partitionierung in Bezug auf funktionale Abhängigkeiten zwischen den Knoten beschrieben wird. Dazu zählt beispielsweise die Gruppierung von militärischen oder zivilen Einsatzkräften.

3.11.2 Hierarchiebildung

Die Wahl der Koordinatoren erfolgt bei HRM stets auf Basis von Knotenprioritäten, welche zwischen den Hierarchielevels unterschiedlich sein können. Die Priorität ist dabei von der Topologie auf dem jeweils darunterliegenden Hierarchielevel abhängig. Dadurch werden Koordinatoren in der Nähe von Knoten mit hoher Konnektivität platziert, sodass die Kommunikationswege innerhalb der Managementinfrastruktur von HRM möglichst kurz und dadurch die Belastung des Netzwerks möglichst niedrig gehalten werden. Andere Protokolle wie *Low-Energy Adaptive Clustering Hierarchy* (LEACH) [118] oder CBR verwenden zufällige Platzierungen ihrer *Cluster Heads.* Diese werden periodisch gewechselt, sodass jeder Knoten eine Koordinatorrolle je Periode zugeordnet bekommen kann. Alternative Ansätze

[119] zur Platzierung eines *Cluster Heads* (Koordinator) für Sensornetzwerke beachten während der Koordinatorplatzierung das Mobilitätsverhalten einzelner Knoten. Dabei wird jener Knoten bevorzugt, welcher für eine festgelegte Periode möglichst wenige Ortswechsel durchläuft und somit eine möglichst stabile Struktur unterstützt. Zusätzlich zu diesem Kriterium verwendet der *Analytical Hierarchy Process* (AHP) [120] die Distanz zwischen einem Knoten und dem jeweiligen Clusterzentrum. Dadurch werden Knoten bevorzugt, welche nahe am Clusterzentrum liegen. Da AHP insbesondere für Sensornetzwerke konzipiert ist, wird die Distanz dabei durch GPS-Koordinaten gebildet. Des Weiteren wird die Platzierung von *Cluster Heads* zentral durch die Basisstation des Sensornetzes durchgeführt und unterscheidet sich somit signifikant von der dezentralen Signalisierung von HRM.

Bei einem Vergleich von HRM mit den heute typischen Routingprotokollen OSPF und BGP fällt wiederum die vorgegebene Konfiguration der Hierarchie auf. Beispielsweis sind die *Areas* von OSPF durch den Netzwerkoperator vorgegeben. Analog dazu geschieht die Konfiguration für BGP. Dabei wird zwischen *External BGP* für ein Inter-AS-Routing und *Internal BGP* für die Verteilung von Routen innerhalb eines AS unterschieden. Im Vergleich zum *Backup Designated Router* beim Einsatz von OSPF, wird bei HRM keine solche Backupinstanz für Koordinatoren erstellt. Bei Erkennung eines Koordinatorausfalls wird durch die Signalisierungen der Kontrollebene automatisch eine neue Instanz platziert. Dieses Vorgehen vermeidet eine kontinuierliche Synchronisation zwischen einer Koordinatorinstanz und einer möglichen Backupinstanz. Falls dabei die Konvergenzzeit der Hierarchie kürzer als die Gültigkeitsdauer der verteilten Routingdaten ausfällt, ist keine Beeinträchtigung des Routing zu erwarten.

Bei PNNI repräsentiert ein *Peer Group Leader* (PGL) auf dem nächsthöheren Hierarchielevel das Netzwerk seiner untergeordneten Mitglieder seiner *Peer Group*. Die Wahl des PGLs ist dabei von den vergebenen Prioritäten sowie den jeweiligen ATM-Adressen der Entitäten des zugrundliegenden ATM-Netzwerks abhängig. Beide werden vom Netzwerkoperator festgelegt, wodurch sich PNNI wiederum von HRM und seiner Art der Prioritätsvergabe unterscheidet. Die Wahl gewinnt bei PNNI stets die Entität mit der höchsten Priorität und ATM-Adresse. Dies ähnelt wiederum dem Konzept von HRM, sodass ebenfalls eine Veränderung des Wahlergebnisses aufgrund veränderten Prioritäten zur Laufzeit des Netzwerks möglich ist.

3.11.3 Adresszuweisung

Die Struktur der HRMIDs entspricht den *Hierarchical Location Identifiers* (HLI) aus [121]. Durch die hierarchische Struktur der Adressen wird eine einfache Zielaggregation ermöglicht, welche bei HRM dafür verwendet wird, um mit einer bekannten Route möglichst viele Zielknoten erreichen zu können. Die für HLI eingeführte Spezifikation für Sensornetzwerke beruht auf der Annahme, dass ein solches Netzwerk zentral geplant wird und somit Adressen per Definition vorgegeben sind, als möglicher Weg einer automatischen Konfiguration wird ausschließlich auf DHCP verwiesen. Dieses Vorgehen steht jedoch im Widerspruch zur Forderung einer autonomen Arbeitsweise von HRM. Das in dieser Arbeit vorgestellte Routingmanagement bedient sich stattdessen der hierarchischen Struktur seiner Managementinstanzen und weist mit deren Hilfe allen Netzwerkschnittstellen im Netzwerk auf autonome Art und Weise explizite Adressen zu.

Typischerweise wird heute für kleinere, lokale Netzwerke auf Protokolle wie DHCP oder BOOTP zurückgegriffen, um automatisch IP-Adressen an hinzukommende Knoten zu verteilen. Dafür muss vom Netzwerkoperator ein IP-Adressbereich festgelegt werden. Dieser automatisierte Ansatz führt zu Skalierungsproblem für sehr große Netzwerke. In diesem Fall werden IP-Adressen für OSPF- und BGP-Router mit Hilfe von spezieller Netzplanungssoftware zentral durch den Netzwerkoperator festgelegt. Beispiele solcher Softwarelösungen sind unter der Bezeichnung *IP Address Management* (IPAM) zu finden.

Die Adressierung von PNNI beruht ebenfalls auf manuellen Eingaben. Die ATM-spezifischen Adressen werden durch den Netzwerkoperator festgelegt. Sie bestimmen sowohl die Netzwerkunterteilung in *Peer Groups* als auch die resultierende Hierarchie zwischen den *Peer Groups*.

In HSR bestehen die verwendeten *Hierarchical IDs* (HIDs) aus kombinierten Adressen des Protokolls von Schicht 2 des OSI-Modells. Dafür können beispielsweise MAC-Adressen verwendet werden. Eine HID besteht somit aus den MAC-Adressen aller Knoten, welche sich entlang des Pfades von der Spitze der Hierarchie bis hin zum jeweils betrachteten Knoten selbst befinden. Da die Erstellung der Hierarchie nur abstrakt definiert ist, kann eine Einschätzung einer möglichen automatischen Konfiguration nicht zuverlässig gegeben werden. Zusätzlich werden bei HSR sogenannte logische Adressen eingesetzt. Diese sind ähnlich zu IP-Adressen zu sehen und bestehen sowohl aus einem Netzwerk- als auch Hostteil. Für die Vergabe wird in der Literatur auf eine Nutzergruppierung anhand gemeinsamer Eigenschaften verwiesen. Als Beispiele werden militärische und zivile Einsatzkräfte aufgeführt. Dies führt zu der Annahme, dass die Gruppierung – somit auch die Netzstrukturierung – auf Basis von manuellen Eingaben erfolgen muss.

Beim Vergleich von HRM mit dem simulierten Routingdienst von FoG ist insbesondere die Gate-orientierte Strukturierung des Netzwerks zu sehen. Jedes Gate ist durch seine eindeutige Nummer identifizierbar. Diese Gatenummern können kombiniert werden, sodass sie einen Pfad zu einem gewünschten Ziel beschreiben. Im Vergleich dazu wendet HRM das Vergabeschema von IP an, sodass jeder Knoten pro Netzwerkschnittstelle eine eigene Adresse zugewiesen bekommt. Ein Pfad zu einem Ziel wird somit durch eine Kombination von Knotenadressen beschrieben.

3.11.4 Verteilung von Routingdaten

Für HRM wird eine strikte Unterteilung in Kontroll- und Datenebene verwendet. Dadurch verbleiben die einzelnen Module von HRM modular und unabhängig voneinander. Die innerhalb der Kontrollebene platzierten Koordinatorinstanzen werden zudem auf unterschiedlichen Hierarchielevels angeordnet. Verwendung finden sie bei der Verteilung von Adressen und Routingdaten. Die dafür notwendigen Signalisierungen erfolgen, aufgrund der bei HRM angewandten Aufteilung des Netzwerks in Cluster, typischerweise in einem örtlich begrenzten Bereich. Des Weiteren werden bei der Verteilung von Routingdaten verschiedene Methoden zur Datenreduktion angewandt. Dies unterstützt die Skalierbarkeit der Kontrollebene und ermöglicht eine Anwendung von HRM für große Netzwerke, wodurch sich das Konzept grundlegend von anderen Routingprotokollen unterscheidet. Beispielsweise werden bei *Dynamic Source Routing* (DSR) [122] im Gegensatz zu HRM die Routingdaten ohne örtliche Begrenzung auf Basis einer flachen Managementstruktur von Knoten zu Knoten verbreitet. Auf Basis dieser Daten ist somit jeder Knoten in der Lage, eine Gesamtroute zu berechnen.

Ähnlich zu OSPF wird bei HRM insbesondere für *RouteReport*-Nachrichten ein *Link State* Protokoll eingesetzt, um Routingdaten zwischen den Knoten zu verteilen. Im Vergleich zu *Distance-Vector- oder Path-Vector*-Protokollen verspricht dies eine schnelle Konvergenzzeit bei Topologieänderungen zum Preis erhöhter Nutzung von Prozessorzeiten und Speicherkapazitäten. Im Gegensatz zu OSPF werden bei HRM alle Routingdaten auf Basis einer automatisch strukturierenden Managementinfrastruktur verteilt. Des Weiteren fällt beim Vergleich zwischen HRM und BGP auf, dass unterschiedliche Detailgrade für die signalisierten Routingdaten verwendet werden. Typischerweise operiert ein BGP-Router im Internet auf Inter-AS-Ebene und ihm werden keine Details über interne Strukturen fremder AS mitgeteilt. Somit besitzt er auch keine Kenntnis über die Hop-Distanz, welche die Anzahl von Knoten beschreibt, die zur Querung des jeweiligen AS passiert werden müssen. Dadurch ist es möglich, dass ein BGP-Router eine Route wählt, welche im Vergleich zu Alternativen eine geringere Anzahl zu querender AS beinhaltet, aber dennoch zu einer unnötig hohen Anzahl zu passierender Knoten führt. Dem wird bei HRM durch die Übermittlung der Hop-Distanzen einzelner cluster-querender Routen in *RouteReport*-Nachrichten entgegen gewirkt, sodass trotz der eingesetzten Topologieaggregation für entfernte Knoten

bekannt ist, ob eine Route im Vergleich zu Alternativen zu einer signifikant erhöhten Anzahl zu passierender Knoten führt.

Ähnlich heutiger Netzwerke verwendet HRM eine knotenorientierte Sicht auf Netzwerke. Vergleicht man diese mit FoG anhand eines gegebenen physikalischen Netzwerks fällt eine Analogie zwischen beiden Welten auf. Ein physikalischer Zwischenknoten wird bei FoG durch drei virtuelle Weiterleitungsknoten abgebildet. Ein direkter Vergleich der Verteilung von Routingdaten zwischen HRM und dem simulierten Routingdienst von FoG kann nicht vorgenommen werden, da letzterer kein Signalisierungsprotokoll beinhaltet. Stattdessen stellt HRM einen geeigneten Signalisierungsansatz für ein QoS-Routing zwischen FoG-basierten Netzwerkknoten dar.

3.11.5 Beachtung von Qualitätsanforderungen beim Routing

Das Routing von HRM geschieht ähnlich heutiger IP-Netzwerke stets dezentral in Form von Einzelentscheidungen auf den Knoten des Netzwerks. Dies unterscheidet HRM von quellbasierten Ansätzen wie DSR. Des Weiteren werden bei HRM-basiertem Routing die Qualitätsanforderungen der Anwendung beachtet. Dabei werden ungenügende Pfade durch das Netzwerk in Abhängigkeit von den vorliegenden Routingdaten vermieden. Diese Topologiedaten werden zudem durch die Kontrollebene von HRM stetig aktuell gehalten, sodass Routingentscheidungen bei Kapazitätsänderungen im Netzwerk angepasst werden. HRM reagiert somit flexibel auf Routen- bzw. Kapazitätsänderungen und kann ebenfalls vollkommen neue Pfade während der Betriebszeit eines Netzwerks lernen.

Im Gegensatz zu HRM beachtet das originäre OSPF- und BGP-basierte Routing primär die Hop-Distanzen bei Routenberechnungen. Die Distanz kann dabei durch verschiedene Konfigurationen des Netzwerkproviders beeinflusst werden und ist nicht nur von den physikalischen Hop-Distanzen abhängig. Die zugehörigen QoS-spezifischen Erweiterungen (bspw. QOFPF, OSPF-TE und QPPB) verwenden zusätzliche Metriken zur Beachtung von unterschiedlichen Qualitätsanforderungen. Dennoch bieten sie nicht die gleiche Funktionalität wie HRM. Im Vergleich dazu bietet PNNI für reine ATM-basierte Netzwerke eine ebenbürtige Lösung und kann bei korrekter Konfiguration der ATM-Switches für ähnliche Szenarien wie HRM eingesetzt werden – dafür sind jedoch umfangreiche manuelle Eingaben notwendig.

Bei HSR wird das Routing implizit durch die erstellte Hierarchie vorgegeben. Sowohl Signalisierungs- als auch Anwendungsdaten werden entlang konstanter Pfade durch das Netzwerk geleitet. Im Gegensatz zu HSR geschieht das eigentliche Routing von Anwendungsdaten bei HRM unabhängig von der Verteilung der Kontrollebene und den daraus resultierenden internen Kommunikationswegen. Dadurch unterscheiden sich HSR und HRM signifikant voneinander. HRM zeigt seine Vorteile insbesondere bei Topologieänderungen, bei denen es sich aufgrund seiner autonom adaptierenden Kontrollebene flexibel anpasst und neue Routen automatisch ermittelt und diese wissen als Routingdaten im Netzwerk verteilt.

Bei der Verwendung von *Probes* werden Qualitätsanforderungen für die Übertragung ebenfalls beachtet. Jedoch treten bei der Ermittlung einer Routingentscheidung Verzögerungen aufgrund der Laufzeiten von *Probe*-Nachrichten auf, dies tritt insbesondere bei der Anwendung des 2.Durchlaufs auf. Ein detaillierter Vergleich ist jedoch an dieser Stelle nicht möglich, da durch *Selective Probing* nicht näher spezifiziert wird, wie die Verteilung von Routingdaten im Netzwerk erfolgt. Eine ähnliche Einschränkung gilt für den simulierten Routingdienst von FoG, da dieser kein eigenes Protokoll für die Verteilung von Routingdaten beinhaltet. Er verwendet bei seinen Entscheidungen stets die kürzeste Route. Sollten dabei notwendige QoS-spezifische Gates fehlen, löst der simulierte Routingdienst deren automatische Erstellung im Transferdienst aus.

3.12 Schlussfolgerungen

In Kapitel 3 wurde das neuartige *Hierarchical Routing Management* (HRM) vorgestellt. Das System besitzt sieben wichtige Vorteile, welche es von Alternativlösungen unterscheidet.

- **Autonome Unterteilung des Netzwerks sowie Adressvergabe:** Als erster Vorteil von HRM ist seine Autonomität zu nennen, welche durch seine drei Prozesse zum Netzwerkmanagement ermöglicht wird. Das **erste Protokoll** beinhaltet eine Netzunterteilung sowie den Aufbau einer hierarchischen Kontrollebene, wobei beides bei topologischen Veränderungen automatisch durch Signalisierungen zwischen den Knoten des Netzwerks aktualisiert wird. Die resultierende Managementinfrastruktur wird durch das **zweite Protokoll** zur automatischen Zuweisung von Adressen verwendet, sodass jeder Netzwerkschnittstelle eines Knotens eine eindeutige HRMID zugeordnet wird. Über diese Identifikation ist er innerhalb des Netzwerks eindeutig als Ziel einer Übertragung auswählbar. Auf Basis der vorliegenden Strukturierung und der verteilten Adressen können die Instanzen der hierarchischen Kontrollebene die im Netzwerk existierenden Routen sowie ihre QoS-spezifischen Eigenschaften beschreiben und diese Informationen als Routingdaten mit Hilfe des **dritten Protokolls** im Netzwerk verteilen. Aus diesen Daten erstellt jeder Knoten seine lokale Routingtabelle, welche bei auftretenden Veränderungen im Netzwerk automatisch durch die Kontrollebene aktualisiert wird. Somit beinhaltet jede Routingtabelle aktuelle Daten über die Topologie des Netzwerks und zusätzlich die für jede Route verfügbaren Kapazitäten. Diese vollständig automatisiert ablaufenden Signalisierungen unterscheiden HRM von bekannten Alternativlösungen, wie beispielsweise OSPF oder BGP. Diese setzen eine vorgegebene Netzwerkunterteilung sowie eine durchgeführte Adresszuweisung für die beteiligten Router voraus, sodass erst auf Basis dieser Konfiguration eine Verteilung von Routingdaten und das letztliche Routing im Netzwerk erfolgen können. Ein Vergleich zwischen HRM und ausgewählten Alternativen ist in Tabelle 3.9 von Abschnitt 3.10.9 zu finden.
- **Beachtung von Qualitätsanforderungen beim Routing:** Als zweiter wichtiger Vorteil von HRM ist seine Datenebene zum Routing von Anwendungsdaten zu sehen. Dabei wird ein *Hop-by-Hop*-Routing umgesetzt, welche sowohl die lokale Routingtabelle als auch die Anforderungen der jeweils sendenden Anwendung eintreffender Pakete als Eingabe verwendet. Folglich werden ungeeignete Pfade frühzeitig vermieden, sodass Engpässe bei der Übertragung von Anwendungsdaten verhindert werden. Dadurch wird eine möglichst gute Dienstqualität einer Anwendung gegenüber dem Nutzer ermöglicht. Diesen Vorteil bietet das herkömmliche BE-Routing in heutigen IP-basierten Netzwerken nicht.
- **Beachtung von aktuellen QoS-spezifischen Routeneigenschaften beim Routing:** Die Kontrollebene verteilt kontinuierlich Routingdaten mit QoS-spezifischen Eigenschaften zu jeder bekannten Route im Netzwerk. Aus diesen Daten erstellt und aktualisiert jeder Knoten seine lokale Routingtabelle. Neben den Qualitätsanforderungen der Anwendung dienen diese Daten als zweite Eingabe für eine Routingentscheidung der Datenebene, sodass bei der Ermittlung des jeweils nächsten Knotens stets die aktuellen QoS-spezifischen Eigenschaften von allen Routen zum letztlich gewünschten Ziel beachtet werden. Auch dieser Vorteil steht bei herkömmlichem BE-Routing nicht zur Verfügung.
- **Fairness im Routing und Ausnutzung von Netzwerkressourcen:** Der Routingalgorithmus der Datenebene wendet eine Kombination aus WSPF- und SWPF-Routing an. Dabei wird die kürzeste Route zum Ziel bevorzugt, sofern diese die Qualitätsanforderungen der Anwendungen erfüllt. Somit werden möglichst wenige parallel ablaufende Routinganfragen beeinflusst oder womöglich blockiert. Sobald die WSPF-Strategie keine geeignete Route mehr für die Übertragung der Anwendungsdaten liefert oder die resultierende Route bereits zu stark ausgelastet ist, verwendet der Routingalgorithmus die SWPF-Strategie. In dem Fall werden alle im Netzwerk vorhandenen Routen beachtet und die Route mit der größten Kapazität ermittelt. Dadurch werden mit steigender Anzahl von Datenströmen die Ressourcen im Netzwerk nach und nach ausgenutzt.
- **Eigenständigkeit:** Als weiterer besonderer Vorteil von HRM ist die erreichte Unabhängigkeit der Signalisierungen von vorhandenen Protokollen von Schicht 3 zu nennen. Dadurch ist das

Routingmanagement sowohl für heutige IPv4/v6-basierte als auch für mögliche zukünftige Netzwerke geeignet. Dies unterscheidet HRM von alternativen Ansätzen wie OSPF und BGP, welche jeweils ein IP-basiertes Netzwerk mit vorgegebener Strukturierung und Adressverteilung voraussetzen.

- **Kompatibilität:** Die Signalisierungen der Kontrollebene können sowohl für IPv4- als auch IPv6-basierte Netzwerke angewendet werden. Dabei ist es vorteilhaft, dass das Adressierungsschema der HRMIDs kompatibel zu IP-Adressen ist und somit das QoS-Routing von HRM ebenfalls für heutige Netzwerke verwendet werden kann. Des Weiteren kann das Routingmanagement ohne Anpassungen auch als Routingansatz für FoG-basierte Netzwerke eingesetzt werden, dies wird in Kapitel 4 anhand einer Implementierung verdeutlicht.
- **Skalierbarkeit:** Als letzter – aber dennoch besonders wichtiger – Vorteil von HRM ist seine Skalierbarkeit für große Netzwerke zu nennen. Dabei spielt vor allem die durch die Kontrollebene ausgeführte Unterteilung des Netzwerks in Cluster eine wichtige Rolle. Des Weiteren wird eine mehrstufige Hierarchie von Managementinstanzen zur Verwaltung der einzelnen Cluster verwendet. Dadurch werden sowohl der Speicheraufwand als auch die Berechnungslast auf die Knoten des Netzwerks verteilt.
  Des Weiteren werden durch die Kontrollebene erkannte Pfade aggregiert und die resultierenden aggregierten Routen während der Signalisierung von Routingdaten genutzt, wodurch das verursachte Signalisierungsaufkommen klein gehalten wird.

4 Implementierung des hierarchischen Routingmanagements

Als Basis der Implementierung wurde die Software *FoGSiEm* [10] eingesetzt. Sie stellt den Demonstrator des in Abschnitt 2.3.3 vorgestellten Forschungsansatzes FoG dar. Dabei war insbesondere vorteilhaft, dass FoGSiEm ein modulares Routing unterstützt, wobei verschiedenste Routingimplementierungen zum Einsatz kommen können. An dieser Stelle konnte HRM als sinnvolle Erweiterung von FoG in die Software integriert werden. Dadurch kann das neuartige Routingmanagement auf Basis der durch FoGSiEm gebotenen Umgebung sowohl für Netzwerksimulationen als auch für reale Hardware eingesetzt werden. Dies ermöglicht es, die Signalisierungen der Ebenen von HRM mit möglichst realitätsnahen Bedingungen zu evaluieren.

Das HRM-Konzept wurde als Erweiterung von FoGSiEm vollständig umgesetzt. Zum Startzeitpunkt der Implementierungsarbeiten im Jahr 2010 existierte bereits eine Basisversion von FoGSiEm. In [99] wurde während des Jahres 2011 der erste Prototyp fertiggestellt. Dieser diente zur Studie der Machbarkeit einer Implementierung und ersten Messungen notwendiger Signalisierungen. Im Jahre 2013 wurde vom Autor dieser Arbeit eine Neuimplementierung begonnen und 2014 abgeschlossen. Während dieser Phase wurde die Konzeption der Signalisierungen zu der in Kapitel 3 beschriebenen Form finalisiert. Die entstandene Implementierung bildete ebenfalls die Basis der quantitativen Evaluierungen von Kapitel 6 und steht zudem der Öffentlichkeit als Open-Source-Software für weitere Vergleichsmessungen zur Verfügung [123]. Der Gesamtumfang beträgt etwa 23000[1] Zeilen Quellcode, welche über 84 Dateien verteilt sind und eine Gesamtgröße von 1,8 MB aufweisen.

Die Erläuterungen dieses Kapitels zur Implementierung starten in Abschnitt 4.1 mit einem Überblick über die Softwarearchitektur. Im anschließenden Abschnitt 4.2 werden implementierungsspezifische Details zur umgesetzten Kontrollebene gegeben. Dazu zählen Erweiterungen des HRM-Konzeptes, welche zur Reduktion auftretender Zwischenlösungen während der Aufbauphase der Kontrollebene integriert wurden. Des Weiteren erklärt dieser Abschnitt, wie allgemeine Signalisierungsnachrichten der Kontrollebene mit Hilfe von FoG-Pakete übertragen werden können. Abschnitt 4.2 schließt mit einer Beschreibung der Speicherung von ermittelten Routingdaten. Nach dieser Beschreibung der Kontrollebene enthält Abschnitt 4.3 die wichtigsten Details zur Implementierung der Datenebene. Dabei werden die Struktur der implementierten Routingtabellen sowie die Ermittlung der FoG-spezifischen Routen zu Nachbarknoten erläutert. Des Weiteren wird beschrieben, wie der FoG-spezifische Ablauf des Routingmanagers und des Routingalgorithmus umgesetzt sind. Im Anschluss wird in Abschnitt 4.4 die integrierte Programmierschnittstelle gegenüber Anwendungen beschrieben. Dies wird nachfolgend in Abschnitt 4.5 für die Realisierung von Testanwendungen verwendet. Zum Ende von Kapitel 4 wird die Implementierung des HRM-Konzeptes mit Bezug zu den zuvor in Abschnitt 4.1.1 aufgestellten Anforderungen zusammenfassend betrachtet.

4.1 Softwarearchitektur

Die Beschreibung der Softwarearchitektur gliedert sich in eine Aufstellung der ursprünglichen Anforderungen an die Implementierung sowie einen Überblick über die erstellten Erweiterungen für die Software FoGSiEm.

4.1.1 Anforderungen

Im Vordergrund der Implementierung muss die Umsetzung aller in Kapitel 3 erläuterten Teile des HRM-Konzeptes stehen. Da das Ergebnis auch für die Validierung und zur Bemessung der Betriebskosten

[1] Dieser Wert wurde mit der Software *cloc* ermittelt, welche unter http://cloc.sourceforge.net/ erhältlich ist.

sowie des Nutzens von HRM dienen soll, müssen zusätzlich die folgenden Anforderungen bei der Erweiterung des Netzwerksimulators beachtet werden:

- **dezentrale Funktionsweise:** Die Implementierung muss als paketbasierte Simulation arbeiten. Synchronisationsmechanismen dürfen ausschließlich auf Basis von knotenlokalen Statusinformationen und dem Austausch von simulierten Signalisierungsnachrichten erfolgen. Dadurch kann die Implementierung ebenfalls für reale Netzwerke eingesetzt werden. Beispielsweise ist dies für den Einsatz der Emulatorfunktion von FoGSiEm notwendig.
- **deterministische Ereignisverarbeitung:** Die Implementierung soll eine zuverlässige Basis für eine experimentelle Evaluierung darstellen. Dafür ist eine korrekte Ereignissynchronisation notwendig, sodass ein deterministisches Ergebnis der Hierarchieerstellung sichergestellt ist.
- **Statistiken für Signalisierungen:** Über die verwendeten Signalisierungspakete sollen Statistiken während der Betriebszeit eines Netzwerks aufgezeichnet und in Dateien gespeichert werden können. Dadurch sollen die Betriebskosten von HRM anhand von quantitativen Ergebnissen verdeutlicht werden.
- **Simulation von BE-Routing:** Es muss möglich sein, durch die Datenebene das Verhalten von BE-Routing zu simulieren, wodurch ein direkter Vergleich zwischen HRM- und BE-basiertem Routing ermöglicht wird.
- **Allgemeingültigkeit:** Die Umsetzung des HRM-Konzeptes muss neben einer Anwendung in der Software FoGSiEm ebenfalls für den Einsatz im Kontext von IP gerüstet sein. Die Implementierung muss hierfür möglichst unabhängig von den Implementierungsdetails von FoGSiEm entwickelt sein.
- **grafische Ausgaben zur Beobachtung:** Der Status der Kontroll- und Datenebene muss während der Betriebszeit eines Netzwerks mit Hilfe von grafische Anzeigen beobachtbar sein. Dies soll insbesondere zum Überprüfen der Abläufe innerhalb der Kontrollebene dienen. Zusätzlich soll dadurch eine Überprüfung der resultierenden Einträge in Routingtabellen ermöglicht werden.
- **Model-View-Controller (MVC):** Innerhalb von FoGSiEm wird das typische MVC-Architekturkonzept [124] angewandt. Dieser Ansatz soll ebenfalls für die HRM-spezifischen Erweiterungen verfolgt werden, sodass eine strikte Unterteilung zwischen Prozessen (und deren Datenverwaltung) und grafische Ansichten gegeben ist. Dadurch werden sowohl die Erweiterbarkeit als auch die Wartbarkeit der Implementierung positiv beeinflusst.

Orthogonal zu den oben genannten Anforderungen muss sich die Implementierung in die Umgebung von FoGSiEm einfügen, sodass Gates und Weiterleitungsknoten von FoG zur Paketweiterleitung verwendet werden.

4.1.2 Erweiterung von FoGSiEm

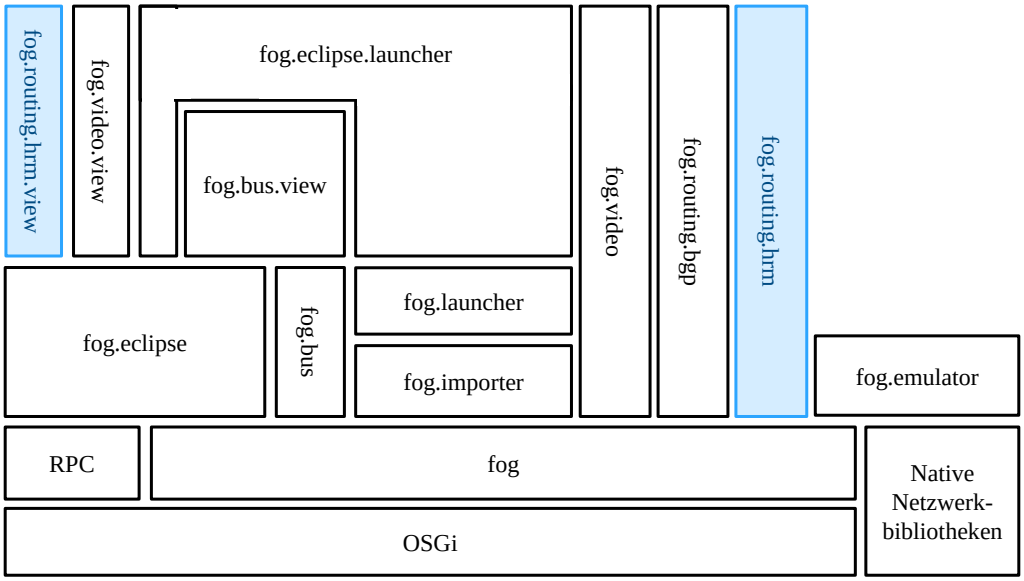

Abbildung 4.1: Erweiterung der FoGSiEm-Softwarearchitektur um einen HRM-basierten Routingdienst

Aufgrund der geforderten Anwendung der MVC-Architektur ergibt sich eine Unterteilung der Implementierung in zwei Plug-Ins für FoGSiEm. Abbildung 4.1 zeigt beide blau hervorgehoben im Kontext der FoGSiEm Softwarearchitektur (siehe Abschnitt 4.2.1 in [93]). Das Plug-In *fog.routing.hrm* beinhaltet alle Softwarekomponenten des HRM-Konzeptes:

- **Kontrollebene (Abschnitt 4.2):** Die Hierarchie der Kontrollebene wird durch die Implementierung autonom für ein gegebenes Netzwerk aufgebaut. Jeder Netzwerkschnittstelle eines Knotens wird mit Hilfe der bestehenden Hierarchie eine Adresse zugewiesen. Auf Basis der vorliegenden Adressen werden existierende Pfade zwischen verschiedenen Knoten und Netzwerkclustern abgeleitet. Die sich daraus ergebenden Routingdaten werden entsprechend Kapitel 3 über die Hierarchie der Kontrollebene kontinuierlich zwischen den Knoten signalisiert. Aus diesen Daten wird auf jedem Knoten eine lokale Routingtabelle abgeleitet, deren Einträge periodisch aktualisiert werden.
- **Datenebene (Abschnitt 4.3):** Die Implementierung nutzt die in Abschnitt 2.3.3 beschriebene Unterteilung in Routingdienst und Transferdienst von FoG aus. Dadurch erhält die Datenebene automatisch Routinganfragen und beantwortet diese mit Hilfe des implementierten Routingalgorithmus entsprechend der Beschreibungen von Abschnitt 3.8. Bei jeder knotenlokalen Routingentscheidung werden dabei einerseits die bei der Routinganfrage mitgelieferten Anforderungen der Anwendung und andererseits die durch die Kontrollebene erstellte Routingtabelle beachtet. Das Ergebnis der Entscheidung wird an die vorhandene FoG-spezifische Paketweiterleitung übermittelt, diese bleibt unverändert erhalten.

Durch das Plugin *fog.routing.hrm.view* stehen verschiedene Möglichkeiten zur Visualisierung des aktuellen Status von HRM zur Verfügung. Mit ihrer Hilfe können die auf jedem Knoten lokal bekannten Daten auf verschiedene Arten grafisch ausgegeben werden, Anhang D enthält dazu Beispiele.

4.1.2.1 Instanzen der Kontrollebene

Auf jedem Knoten wird die Anwendung *HRM-Controller* gestartet. Ihre Implementierung entspricht dem Entwurfsmuster *Singleton* [124] der Softwaretechnik.

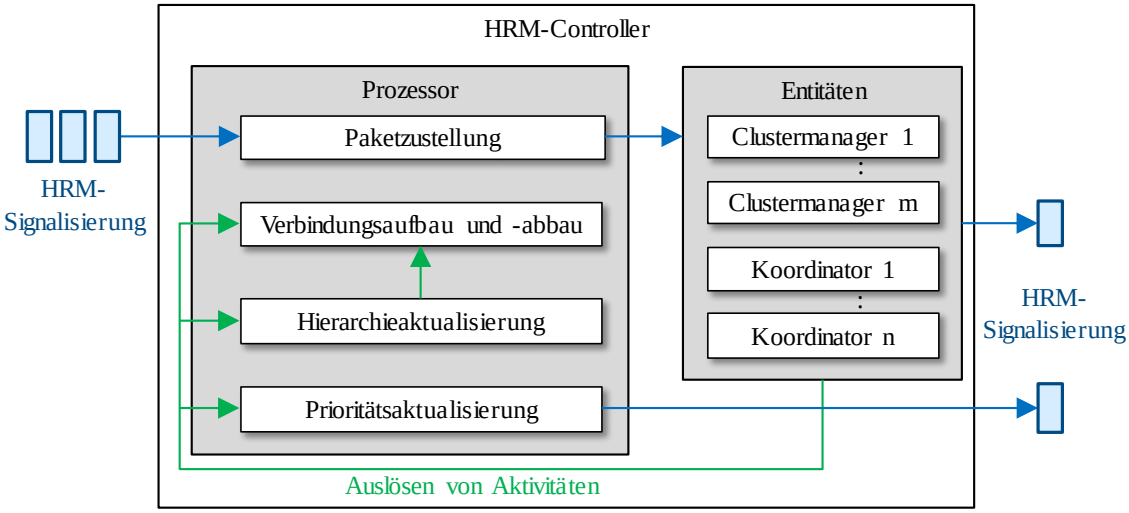

Abbildung 4.2: Der Aufbau und die prinzipiellen Abläufe einer Instanz eines *HRM-Controllers*

Abbildung 4.2 zeigt den prinzipiellen Aufbau eines *HRM-Controllers*. Er beinhaltet einen zentralen Prozessor auf der linken Seite sowie die knotenlokalen Clustermanager und Koordinatoren der Kontrollebene auf der rechten Seite. Jede dieser Entitäten wird automatisch aufgrund von eingehenden Signalisierungsnachrichten durch den *HRM-Controller* nach dem Entwurfsmuster *Factory Method* [124] der Softwaretechnik erstellt. Jede der erstellten Entitäten beinhaltet dabei die jeweils zugehörige Prozesslogik und das Datenmanagement entsprechend der Beschreibungen aus Kapitel 3.

Innerhalb von Abbildung 4.2 sind zusätzlich die Daten- und Kontrollflüsse mit blauen bzw. grünen Pfeilen dargestellt. Der Prozessor ist dabei als eigenständiger Thread umgesetzt, der folgende Aufgaben kontinuierlich abarbeitet:

- **Paketzustellung:** Wie in Abbildung 4.2 anhand der blauen Pfeile erkennbar werden eintreffende Pakete entgegen genommen und auf Basis der vorhandenen Zieladressierung (vgl. Abschnitt 3.3.3) an die jeweilige lokale Entität weitergeleitet. Diese verarbeitet das Paket und kann eine der nachfolgenden Aktivitäten des Prozessors auslösen.
- **Verbindungsaufbau und -abbau:** Der Start und Stopp ausgehender Verbindung geschieht mit Hilfe des zentralen Prozessors. Lokale Entitäten können eine solche Aktion explizit auslösen. Diese Zentralisierung der Verwaltung von Verbindungen vereinfacht die Synchronisation zwischen lokalen Entitäten.
- **Hierarchieaktualisierung:** Sollten externe Koordinatoren wegfallen oder neue hinzukommen, wird durch die erkennende Entität eine Reaktion durch den Prozessor zentral ausgelöst. Dabei versendet er für jeden neu erkannten Koordinator im Namen des jeweils lokalen Clustermanagers eine Clusteranfrage in Form einer *RequestClusterMembership*-Nachricht. Gegebenenfalls wird vor dieser Signalisierung ein Verbindungsaufbau zum Zielknoten gestartet.
- **Prioritätsaktualisierung:** Prioritätswerte eines Knotens werden pro Hierarchielevel innerhalb des Prozessors zentral verwaltet. Entsprechend den Abschnitten 3.3.2 und 3.3.4 wird jede Veränderung ausgewählten anderen Entitäten mit Hilfe expliziter Signalisierungen mitgeteilt.

Die Zentralisierung der aufgeführten Aufgaben in einem eigenständigen Thread pro Netzwerkknoten ermöglicht die geforderte Synchronisation nebenläufiger Prozesse der knotenlokalen Entitäten der Kontrollebene. Dadurch liefert die Implementierung ein deterministisches Ergebnis der Konvergenz der Kontrollebene, sodass sie eine zuverlässige Basis für die Messungen von Kapitel 6 bildet.

4.1.2.2 Instanzen der Datenebene

Die zentrale FoG-Verwaltung startet automatisch beim Start eines Knotens eine Instanz des HRM-spezifischen Routingdienstes und somit eine knotenlokale Instanz der HRM-Datenebene.

4.2 Kontrollebene

Innerhalb der nächsten Abschnitte werden ausgewählte Teile der implementierten Kontrollebene erläutert. Dabei wird insbesondere in Abschnitt 4.2.2 auf implementierungsspezifische Erweiterungen des Konzeptes aus Kapitel 3 eingegangen, welche eine möglichst schnelle Konvergenz bei Hierarchieaufbau und -umbau sicherstellen. Das ursprüngliche Konzept bleibt dabei jedoch bestehen.

4.2.1 Erkennung von Nachbarknoten

Die Software FoGSiEm bietet Mechanismen, welche automatisch neue Nachbarknoten an den lokalen HRM-Routingdienst melden. Sollte ein Nachbarknoten ausfallen, wird dies ebenfalls von FoG erkannt und dem Routingdienst mitgeteilt. Des Weiteren bietet FoGSiEm bereits Mechanismen zur Festlegung und Abfrage der Verzögerung einzelner Links. Die kontinuierliche Erkennung von Nachbarknoten sowie die Abfrage von Linkeigenschaften sind somit innerhalb der Implementierung nicht auf Basis von *AnnounceNeighborNode*-Nachrichten umgesetzt. Stattdessen verwendet die Implementierung die vorhandene Schnittstelle von FoGSiEm. Dieser Kompromiss war bezüglich der gewünschten Allgemeingültigkeit der Implementierung an dieser Stelle notwendig, um HRM als Routingdienst für FoG nutzen zu können. Bei einer Migration des Quellcodes auf ein reines IP-basiertes Netzwerk muss dies beachtet werden.

4.2.2 Reduktion der Konvergenzzeit

Nach dem Start des Netzwerks oder bei Topologieänderungen wird die Kontrollebene entsprechend der Beschreibungen von Abschnitt 3.3 aufgebaut bzw. umstrukturiert. Dies geschieht durch Austausch von Signalisierungsnachrichten, sodass sich die Statusdaten der Knoten stetig ändern. Erst bei Erreichen einer stabilen Hierarchie stoppt dieser Prozess. Die dafür benötigte Zeit ist die Konvergenzzeit der Kontrollebene. Sie sollte möglichst kurz ausfallen. Plötzliche Ausfälle von Knoten, und somit auch von Entitäten, können sie jedoch negativ beeinflussen. Zu diesem Zweck müssen solche Fälle möglichst schnell erkannt und kompensiert werden.

In Abschnitt 3.3.7 wird detailliert erläutert, wie Koordinatorausfälle durch die Kontrollebene auf Basis von Timeouts erkannt werden. Des Weiteren wird im Kontext von HRM eine angepasste Variante von TCP zum Transport von Signalisierungsnachrichten der Kontrollebene eingesetzt. Abschnitt 3.6.1.2 gibt Details zum verwendeten Nachrichtenformat für den Transport von Signalisierungen der Kontrollebene. Dadurch werden Verbindungen bei Ausfall eines Kommunikationspartners automatisch nach einer definierten Zeit (ähnlich TCP-Verbindungen) beendet.

In den nachfolgenden Abschnitten werden implementierungsspezifische Erweiterungen von HRM vorgestellt, welche zur Verringerung der Konvergenzzeit beitragen. Dadurch wird eine schnelle Erkennung und Kompensation von Ausfällen unterstützt.

4.2.2.1 Explizite Signalisierung des Kommunikationsstatus

Für den Fall, dass ein Kommunikationspartner explizit als ungültig markiert und von der Kontrollebene entfernt wird, sind explizite Signalisierungen sinnvoll. Sie ermöglichen eine sofortige Aktualisierung des Kommunikationsstatus. Zu diesem Zweck werden in der Implementierung zusätzliche Nachrichtentypen verwendet:

- **Bestätigung der Kommunikationsanfrage:** Wenn ein Clustermanager eine Kommunikation zu einem untergeordneten Koordinator aufbaut, sendet er ihm entsprechend Abschnitt 3.3.4.3

eine *RequestClusterMembership*-Nachricht. Die Implementierung sieht vor, dass ein Koordinator die Kommunikation erst durch eine *RequestClusterMembershipAck*-Nachricht bestätigen muss, bevor die Kommunikation fortgesetzt und der Koordinator in die Wahl übergeordneter Clustermanager einbezogen wird. Dies geschieht nur, wenn der Koordinator zum Zeitpunkt des Eintreffens der Nachricht weiterhin gültig ist.

- **Beendigung der Kommunikation durch einen Koordinator:** Sollte ein Koordinator als ungültig markiert werden, sendet er vor seiner endgültigen Löschung *InformClusterLeft*-Nachrichten an jeden zugehörigen übergeordneten Clustermanager. Dadurch wird unmittelbar ein Stopp der jeweiligen Kommunikation ausgelöst.
- **Beendigung der Kommunikation durch einen Clustermanager:** Wird ein Clustermanager aufgrund von Hierarchieumstrukturierungen als ungültig markiert, sendet er vor seiner endgültigen Löschung *InformClusterMembershipCanceled* Nachrichten an alle untergeordneten Koordinatoren. Dieser Nachrichtentyp fungiert als Gegenstück zu *InformClusterLeft* und teilt explizit die Beendigung der jeweiligen Kommunikation mit.

Als Folge der expliziten Signalisierungen ergibt sich eine Verringerung der Konvergenzzeit gegenüber einer ausschließlich auf Timeouts basierenden Erkennung von ungültig gewordenen Entitäten.

4.2.2.2 Explizite Validierung von Kommunikationskanälen

Bleiben *AnnounceCoordinator*-Nachrichten von einem entfernten Koordinator aus, kann von einem Knoten- oder Linkausfall ausgegangen werden. In beiden Fällen stehen die Entitäten des jeweiligen Knotens der Kontrollebene nicht mehr zur Verfügung. In Abschnitt 3.3.7 wird eine Gültigkeitsdauer pro Kommunikationskanal eingeführt, welche eine Erkennung der beschriebenen Ausfälle auf Basis von Timeouts realisiert. Zusätzlich zu dieser Art der Erkennung verwendet die Implementierung eine explizite Überprüfung der Existenz entfernter Entitäten. Sie besteht aus einer Request-Response Signalisierung mit Hilfe der zusätzlichen Nachrichtentypen *PingPeer* und *Alive*:

1. Bei Ausfall eines entfernten Koordinators, sendet der erkennende Knoten an den Knoten des Koordinators mit Hilfe der bestehenden Kommunikationskanäle *PingPeer*-Nachrichten.
2. Jeder empfangende Knoten antwortet mit einer *Alive*-Nachricht und signalisiert somit, dass die betroffene Entität weiterhin lebendig ist und für den Wahlalgorithmus zur Verfügung steht. Sollte dies nicht der Fall sein, bleibt die *Alive*-Nachricht aus und die Entität am anderen Ende des Kommunikationskanals wird als ungültig angenommen. Betroffene lokale Statusdaten werden entsprechend aktualisiert.

Diese Vorgehensweise reduziert die Zeit bis zur Löschung ungültig gewordener Kommunikationskanäle und trägt dadurch ebenfalls zur Reduktion der Konvergenzzeit der Kontrollebene bei.

4.2.2.3 Explizite Signalisierung der Koordinatorlöschung

Da die Konvergenzzeit der Strukturierung der Kontrollebene eine wichtige Rolle für die Gesamtperformanz des Systems darstellt, ist es sinnvoll, erkannte Verdrängungen und die nachfolgende Entfernung eines existierenden Koordinators ebenfalls an umliegende Knoten zu signalisieren. Zu diesem Zweck wird der Nachrichtentyp *InvalidateCoordinator* innerhalb der Implementierung verwendet. Sobald ein Koordinator ungültig geworden ist, werden einmalig Nachrichten dieses Typs verbreitet. Die Ausbreitung erfolgt dabei analog zu den *AnnounceCoordinator*-Nachrichten, sodass sie ausschließlich Knoten im Clusterradius r des jeweiligen Hierarchielevels erreichen. Diese erhalten somit Kenntnis darüber, dass ein ehemals bekanntgegebener Koordinator nicht mehr zur Verfügung steht. Die Zuordnung zwischen Bekanntgabe und Rücknahme geschieht dabei auf jedem Empfänger auf Basis der Entität-ID und Knoten-ID des Senders.

Die *InvalidateCoordinator*-Signalisierungen sind eine optionale Ergänzung zu den *Leave/Resign*-Nachrichten. Durch ihre Verwendung wird die Konvergenzzeit während der (Re-)Strukturierung der Kontrollebene zusätzlich reduziert. Würde man diese Nachrichten nicht verwenden, würde die Entfernung einer Koordinatorinstanz erst nach Ausbleiben von *AnnounceCoordinator*-Nachrichten – und somit nach Ablauf einer definierten Zeit – erkannt werden. Insbesondere für höhere Hierarchielevels spielt die eingeführte Signalisierung eine wichtige Rolle, da hier der Verdrängungsfall (und damit auch die Entfernung einer höheren Koordinatorinstanz) im Vergleich zu L0-Koordinatoren aufgrund der eher verzögerten Kommunikation und Ermittlung von Prioritäten häufiger auftritt.

4.2.3 Koordinatorenwahlen

Insbesondere die Synchronisation von Ereignissen während einer Koordinatorwahl ist bei der Implementierung des HRM-Konzeptes von besonderer Herausforderung. Es ist sinnvoll, die notwendigen atomaren Verarbeitungsschritte zu identifizieren und diese möglichst als Einzelfunktionen zu kapseln. Diese ist Bestandteil der HRM-Implementierung.

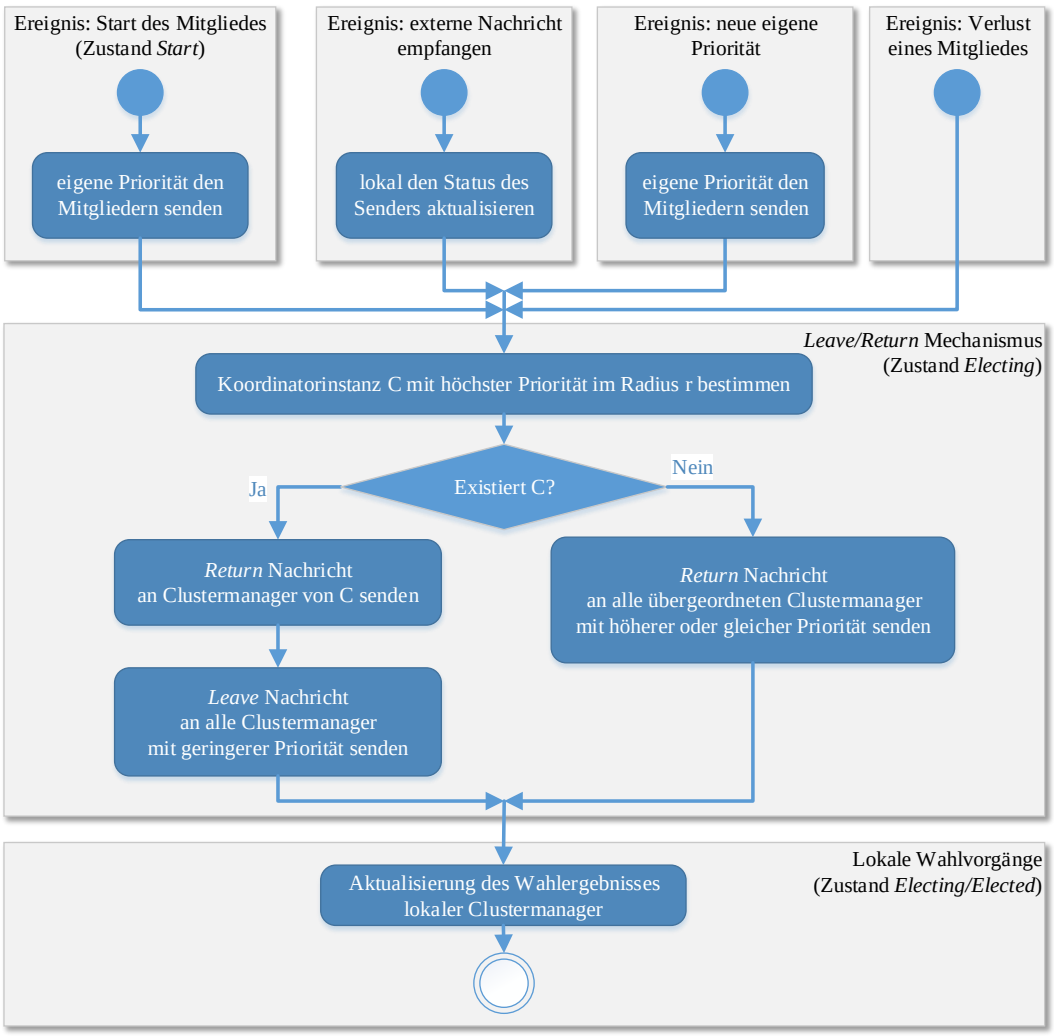

Abbildung 4.3: Ereignisverarbeitung eines Wahlmitgliedes

Abbildung 4.3 zeigt die resultierenden Abläufe von Phase 1 und 2 der Hierarchieerstellung für ein Wahlmitglied. Dies kann sowohl ein Koordinator als auch ein Clustermanager sein. Da sich während eines Hierarchieaufbaus bzw. -umbaus die Prioritäten verändern, wird die Abarbeitung der dargestellten Schritte kontinuierlich wiederholt. Dieser Vorgang endet sobald sich die Prioritäten nicht mehr ändern und die Hierarchieerstellung zu einer stabilen Lösung konvergiert ist. Die Auslösung der einzelnen

Schritte geschieht dabei prinzipiell durch eines der vier folgenden Ereignisse durch den knotenlokalen *HRM-Controller* und seinen Prozessor:

- **Start des Wahlmitgliedes:** Ein neuer Clustermanager bzw. eine Wahlmitgliedschaft eines Koordinators bei einem übergeordnetem Clustermanager wird gestartet. Im ersten Schritt wird die eigene Priorität den jeweils anderen Wahlmitgliedern mitgeteilt.
- **Externe Nachricht empfangen:** Von einem anderen Wahlmitglied wird eine Nachricht empfangen. Dabei beeinflussen folgende Nachrichtentypen das lokale Wahlergebnis:
  - **Für Hierarchielevel 0:** *PriorityUpdate*
  - **Für Hierarchielevel 1 und höher:** *PriorityUpdate, Winner, Resign, Leave, Return*
- **Neue eigene Priorität:** Der lokale *HRM-Controller* ermittelt eine neue Priorität für das jeweilige Hierarchielevel. Diese wird umgehend den anderen Wahlmitgliedern mitgeteilt.
- **Entfernung des Wahlmitgliedes:** Das Wahlmitglied wird als ungültig markiert und gelöscht. Dies kann sowohl ein Clustermanager als auch eine Koordinatorinstanz sein.

Jedes dieser Ereignisse wird durch zwei Schritte beantwortet:

1. ***Leave/Return*-Mechanismus:** Es werden die Zugehörigkeiten zu Wahlmitgliedschaften mit Hilfe des DCE-Algorithmus aus den Abschnitten 3.3.4.6 und 3.3.4.7 festgelegt. Für Phase 2 werden dabei an übergeordnete Clustermanager einzelne *Leave*- bzw. *Return*-Nachrichten versendet, sodass jeder Koordinator nur bei demjenigen eine aktive Wahlmitgliedschaft besitzt, der für das jeweilige Hierarchielevel die Koordinatorinstanz mit der höchsten Priorität im Radius *r* aufweist. Sollte auf dem Level kein Clustermanager mit aktiver Koordinatorinstanz existieren, werden bei allen übergeordneten Clustermanagern mit höherer oder gleicher Priorität die Wahlmitgliedschaften (re-)aktiviert.
2. **Wahlvorgänge:** Anschließend an die *Leave/Return*-Signalisierungen werden alle lokalen Wahlvorgänge aktualisiert. Der detaillierte Ablauf dieses Schrittes entspricht den Ausführungen von Abschnitt 3.3.2.2 für Phase 1 und Abschnitt 3.3.4.4 für Phase 2.

Nach Auslösen dieser Reaktionen auf das eingetroffene Ereignis kehrt die Kontrolle zurück an den Aufrufer, sodass der Prozessor des knotenlokalen *HRM-Controllers* mit der Abarbeitung weiterer Aufgaben fortfahren kann.

4.2.4 Periodische Signalisierungen

Für periodische Aufgaben einzelner Entitäten verwendet die Implementierung die zentrale Ereignissimulation der Software FoGSiEm. Mit ihrer Hilfe ist der Start einzelner Funktionen zu festgelegten Zeitpunkten möglich, sodass periodische Aktualisierungen durch *AnnounceCoordinator*- sowie *RouteReport/RouteShare*-Nachrichten signalisiert werden. Letztere sind für die in Abschnitt 3.5 beschriebene kontinuierliche Aktualisierung von Routingdaten im Netzwerk notwendig.

4.2.5 Verwendung von FoG-Paketen

Innerhalb der Implementierung in FoGSiEm werden FoG-Pakete sowohl für Anwendungsdaten als auch für die Übermittlung von Signalisierungsnachrichten der Kontrollebene eingesetzt[2].

[2] Analog zur Einbettung in FoG-Pakete ist ebenfalls eine Einbettung der HRM-Signalisierungsdaten in *Ethernet Frames* möglich. Weitere Details sind dazu in Anhang B.4 gegeben.

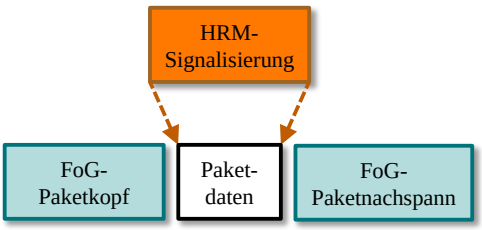

Abbildung 4.4: Transport von Signalisierungsdaten der Kontrollebene mit Hilfe von FoG-Paketen

Wie in Abbildung 4.4 dargestellt wird eine Nachrichten der Kontrollebene als Daten eines FoG-Pakets – kodiert zwischen dem Kopf und dem Nachspann – übertragen. Die Bestimmung der notwendigen Werte für den FoG-Paketkopf sowie den FoG-Nachspann erfolgt dabei mit Hilfe der existierenden FoG-spezifischen Mechanismen, wie sie durch FoGSiEm bereitgestellt werden. Dabei wird der Zielknoten im Paketkopf mit Hilfe der jeweiligen Knoten-ID eindeutig festgelegt, welche mit Hilfe des HRM-spezifischen Routingdienstes bestimmt werden können.

4.2.6 Hierarchical Routing Graph

Jeder *HRM-Controller* verwendet für seinen Knoten eine Instanz eines *Hierarchical Routing Graph* (HRG) zur Speicherung des jeweils lokal vorhandenen Wissens über existierende Pfade im Netzwerk. Diese Daten dienen als Eingabe des Prozesses zur Verteilung von Routingdaten im Netzwerk. Die Implementierung basiert dabei auf der externen Bibliothek *Jung* [125]. Sie stellt die allgemeine Graphenverwaltung sowie eine Schnittstelle zur Verwendung des *Dijkstra*-Algorithmus für Routenberechnungen zur Verfügung. Ein HRG besteht dabei prinzipiell aus HRMIDs, welche über unidirektionale Links miteinander verbunden sind. Eine HRMID in einem HRG kann entweder eine Netzwerkschnittstelle oder einen Cluster identifizieren. Dadurch ergeben sich Links auf unterschiedlichen Hierarchielevels. Der Graph ist somit nicht zusammenhängend und enthält alle aggregierten Sichtweisen auf die Netzwerktopologie, welche dem jeweiligen Knoten bekannt sind. Ein Link kann somit eine der folgenden Entsprechungen haben:

- eine **knotenlokale Verbindung** zwischen zwei HRMIDs (Sie können entweder zu Netzwerkschnittstellen des gleichen Knotens gehören oder es handelt sich dabei um die Adressen von zwei direkt benachbarten Clustern.)
- eine **direkte Verbindung** zwischen zwei benachbarten Knoten
- eine **aggregierte Route** zwischen zwei entfernten Knoten

Für die korrekte Routenberechnung entsprechend des HRM-Konzeptes müssen zudem folgende zusätzlichen Attribute für jeden Link gespeichert werden:

- **Datenrate:** Der Wert stellt die maximal verfügbare Datenrate entlang dieses Links dar.
- **Verzögerung:** Die minimal zu erwartende Verzögerung bei der Benutzung dieses Links muss für die Bestimmung der Gesamtverzögerung einer ermittelten Route bekannt sein.
- **Anzahl von Hops:** Um die Hop-Distanz für eine ermittelte Route bestimmen zu können, muss für jeden gespeicherten Link die Anzahl Hops gespeichert werden.
- **Auslastung:** Entsprechend Abschnitt 3.8.2 muss die Auslastung einzelner Links zur Bestimmung der Gesamtauslastung einer ermittelten Route bekannt sein.
- **Timeout:** Die Gültigkeit jedes gespeicherten Links ist entsprechend Abschnitt 3.5.4.3 zeitlich begrenzt. Zu diesem Zweck wird die absolute Zeit zur Löschung des Links gespeichert.

Ein Knoten erhält die für seinen lokalen HRG notwendigen Routingdaten durch folgende Quellen:

- **knotenlokale Links:** Alle Netzwerkschnittstellen eines Knotens sind knotenlokal über einen logischen Link miteinander verbunden. Für diese Art Links werden in der Implementierung besondere Annahmen verwendet. Sie werden nachfolgend in Abbildung 4.5 erläutert.
- ***AnnounceNeighborNode*-Nachrichten:** Durch diese Signalisierung werden lokale Nachbarknoten identifiziert. Für die Links zu ihnen werden die QoS-Eigenschaften in Abhängigkeit von den lokal bekannten Linkeigenschaften und vorliegenden Reservierungen abgespeichert.
- ***RouteReport/RouteShare*-Nachrichten:** Durch den allgemeinen Prozess zur Verteilung von Routingdaten werden zusätzliche Links in einem HRG eingetragen. Die QoS-Eigenschaften eines solchen Links werden der jeweiligen Signalisierung entnommen. Der Detailgrad eines HRG wird ausschließlich durch die Clusterunterteilung des Netzwerks und der daraus resultierenden Signalisierung von Routingdaten beeinflusst. Beispielsweise enthält der HRG des Knotens mit instanziiertem TOP-Koordinator eine abstrakte Sicht über das Gesamtnetzwerk sowie die jeweils signalisierte Sicht untergeordneter Koordinatoren.

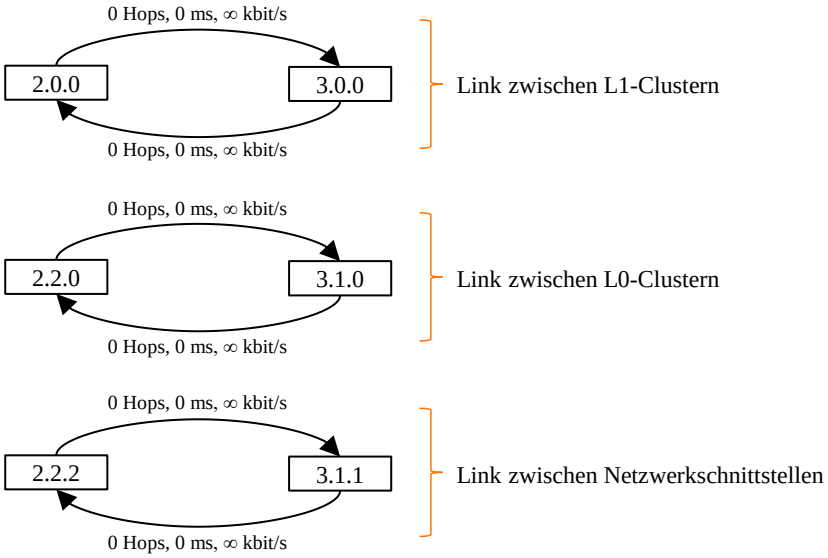

Abbildung 4.5: Beispiel eines HRG für einen Link zwischen knotenlokalen Netzwerkschnittstellen

Abbildung 4.5 zeigt einen ausgewählten Teil des HRGs von Knoten 5 aus dem Beispielszenario von Kapitel 3. Darin sind die resultierenden Links enthalten, welche aufgrund der zwei knotenlokalen Netzwerkschnittstellen mit den HRMIDs 2.2.2 und 3.1.1 im HRG gespeichert werden. Neben dem Link zwischen beiden Netzwerkschnittstellen gehören dazu ebenfalls die Links für die übergeordneten L0- und L1-Cluster. Alle Links besitzen dabei die gleichen QoS-Eigenschaften:

- **Hop-Distanz ist 0:** Da der Link zwischen zwei knotenlokalen Netzwerkschnittstellen besteht, wird seine Distanz mit 0 festgelegt.
- **Verzögerung von 0 ms:** Die knotenlokale Verzögerungen während der Paketweiterleitung wird auf 0 ms festgelegt und somit vernachlässigt.
- **Unbegrenzte Datenrate:** Die lokale Paketzustellung wird stets als unmittelbar angenommen und somit die lokale Wartezeit eines Pakets vernachlässigt. Daher wird die lokale Datenrate eines Routers nicht als limitierender Faktor angesehen und als „unendlich“ innerhalb der Implementierung dargestellt.

Es ist aus Abbildung 4.5 zu erkennen, dass ein HRG nicht zusammenhängend sein muss. Auf Basis seiner gespeicherten Links können Routen in Abhängigkeit des Hierarchielevels bestimmt werden. Dies wird ausgenutzt, um die Routingdaten für ausgehende *RouteReport*- und *RouteShare*-Nachrichten in

Abhängigkeit vom betrachteten Hierarchielevel zu ermitteln. Die Implementierung des *HRM-Controllers* verwendet den typischen *Dijkstra*-Algorithmus, um innerhalb eines HRGs die jeweils kürzeste Route zwischen zwei HRMIDs zu berechnen. Durch Deaktivierung von bereits genutzten ausgehenden Links der Quelle und erneuter Ausführung des *Dijkstra*-Algorithmus werden weitere Routen im HRG ermittelt. Dabei wird für jede Route der Link mit der minimal verfügbaren Datenrate bestimmt und dieser Wert als resultierende maximale Datenrate verwendet. Die Auslastung der Gesamtroute ergibt sich dagegen durch Ermittlung der maximalen Auslastung der verwendeten Links. Zur Bestimmung der Gesamtverzögerung einer Route werden alle Einzelverzögerungen der verwendeten Links aufsummiert. Anhang D.2 zeigt ein Beispiel eines vollständigen HRGs.

4.3 Datenebene

Innerhalb der nachfolgenden Abschnitte werden ausgewählte Teile der implementierten Datenebene erläutert. Dabei werden die für das Ermitteln und Speichern von Routen notwendigen Datenstrukturen beschrieben. Zusätzlich wird die Umsetzung des Routingmanagers sowie des Routingalgorithmus beschrieben.

4.3.1 Routingtabellen

Wie in Abschnitt 3.5 erläutert, werden die Daten der lokalen Routingtabelle eines Knotens aus der Exploration direkter Nachbarschaftsbeziehung sowie aus empfangenen *RouteShare*-Nachrichten entnommen. Prinzipiell wird dabei zwischen drei Typen von Einträgen unterschieden:

- **Lokale Routen:** Jeder Knoten kann beliebig viele Netzwerkschnittstellen besitzen, jeder wird durch die Kontrollebene eine HRMID zugewiesen. Für jede dieser HRMIDs wird eine lokale Route gespeichert.
- **Nachbarrouten:** Aus der Erkennung direkter Nachbarn und der Bestimmung der Eigenschaften des jeweiligen Links zu ihnen wird jeweils mindestens ein Eintrag in der lokalen Routingtabelle hinterlegt. Sollte ein Knoten aufgrund der Signalisierungen aus Abschnitt 3.5.1 feststellen, dass ein Nachbar ebenfalls HRMIDs fremder übergeordneter Cluster besitzt, wird pro Cluster ein zusätzlicher Eintrag angelegt.
- **Routen zu entfernten Knoten:** Diese Routen werden typischerweise aus *RouteShare*-Nachrichten gewonnen und beschreiben Routen zu entfernten Zielen im Netzwerk.

Aufgrund der Konzeption beinhaltet ein Eintrag einer Routingtabelle in der Implementierung folgende Werte[3]:

- **Ziel:** die HRMID des Zielknotens bzw. -clusters
- **Nächster Hop:** die HRMID des nächsten Knotens in Richtung des Ziels
- **Hop-Distanz:** die Anzahl Zwischenknoten zum Ziel
- **Auslastung:** die relative Auslastung der Route in Prozent
- **Verzögerung:** die zu erwartende minimale Verzögerung entlang der Route
- **Datenrate:** die maximal mögliche Datenrate entlang der Route
- **Datenrate des Links zum nächsten Hop:** die Datenrate des Links zum nächsten Knoten
- **Timeout:** die absolute lokale Zeit, nach welcher die Route automatisch entfernt wird

Es ist daraus ersichtlich, dass pro Eintrag immer eine absolute Zeit zur automatischen Löschung gespeichert wird. Nur durch die kontinuierlichen Prozesse aus Abschnitt 3.5.4 zur periodischen Aktualisierung

[3] Weitere Werte werden durch die Implementierung zur besseren Nachvollziehbarkeit der Signalisierungen gespeichert. Sie werden in Anhang D.3 anhand einer ausführlichen Routingtabelle erläutert.

von Routingtabelleneinträgen wird eine solche Löschung verhindert und sichergestellt, dass existierende Routen in den Routingtabellen verbleiben.

Innerhalb der Implementierung wurde die Datenrate zum nächsten Knoten als zusätzlicher Wert für jeden Routingtabelleneintrag eingeführt. Sollten für ein Ziel mehrere ausgehende Routen bekannt sein, dient der Wert als zusätzliches Kriterium bei der Ermittlung einer Routingentscheidung. Dies ist insbesondere bei schnell aufeinander folgenden Reservierungen interessant. In diesem Fall liegt dieser Wert für jede Route aufgrund des lokalen Wissens über direkte Nachbarn (siehe Abschnitt 3.5.1) sowie den automatischen Aktualisierungen nach erfolgten Reservierungen (siehe Abschnitt 3.7) unmittelbar vor. Er reduziert somit den Einfluss von Verzögerungen während der Signalisierung von Routingdaten.

4.3.2 Neighbor Routing Graph

Zusätzlich zum HRG speichert der *Neighbor Routing Graph* (NRG) lokales Wissen eines Knotens über die existierende Topologie des Netzwerks. Dabei bildet ein NRG die notwendige Ergänzung zur jeweiligen Routingtabelle, sodass Anwendungsdaten auf Basis der FoG-spezifischen Netzstrukturen zum jeweils nächsten Nachbarknoten in Richtung des Ziels übertragen werden können. Zu diesem Zweck ordnet ein NRG jedem Nachbarknoten bekannte FoG-Routen zu. Jede Route startet und stoppt dabei auf dem zentralen FoG-Weiterleitungsknoten des jeweiligen Quell- bzw. Zielknotens. Entsprechend Abschnitt 2.3.4.5 werden innerhalb eines NRG die verwendeten Gates und Weiterleitungsknoten mit Hilfe von Gatenummern bzw. Knoten-IDs eindeutig identifiziert.

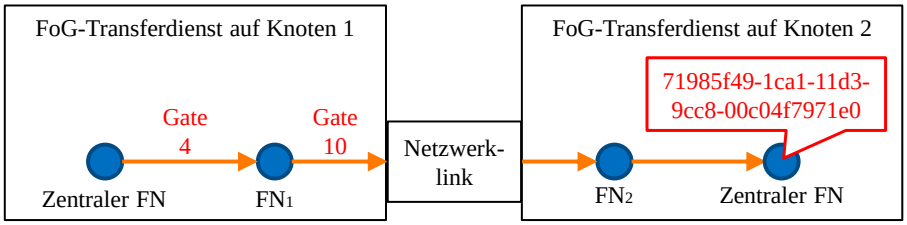

Abbildung 4.6: Beispiel einer Route zwischen zwei Nachbarknoten

Abbildung 4.6 zeigt ein Beispiel einer FoG-Route zu einem Nachbarknoten. Die in der Abbildung dargestellten Gatenummern und Bezeichner sind zufällig gewählt. Knoten 1 gewinnt die zur Speicherung dieser Route notwendigen Informationen mit Hilfe von *AnnounceNeighborNode*-Nachrichten aus Abschnitt 3.3.1:

- **Route zum Nachbarknoten:** Knoten 2 teilt innerhalb seiner *AnnounceNeighborNode*-Nachrichten an Knoten 1 die Knoten-ID seines lokalen Weiterleitungsknotens FN_1 mit. Dieser ist der einzige für Knoten 1 unmittelbar bekannte Weiterleitungsknoten von Knoten 2. Aufgrund der allgemeinen Signalisierungen von FoG kennt die HRM-Routingdienstinstanz auf Knoten 1 bereits die Route zu FN_2. Sie besteht aus Gate 4 und 10.
- **Zentrale Knoten-ID des Nachbarknotens:** Innerhalb seiner *AnnounceNeighborNode*-Nachrichten teilt Knoten 2 ebenfalls die Knoten-ID seines zentralen Weiterleitungsknotens mit.

Knoten 1 ist somit in der Lage, die FoG-spezifische Route zum Nachbarknoten 2 in seinem lokalen NRG zu speichern. Sie besteht aus den zwei FoG-typischen Routensegmenten: „[4, 10]“ und „[71985f49-1ca1-11d3-9cc8-00c04f7971e0]“. Die im ersten Teil enthaltene Liste aus Gatenummern beschreibt einen expliziten Pfad über FN_1 zu FN_2. Dieser Weiterleitungsknoten befindet sich bereits innerhalb des Transferdienstes von Knoten 2. Das nachfolgende Zielsegment beinhaltet die Knoten-ID des nächsten Ziels: der zentrale Weiterleitungsknoten von Knoten 2. Der inkrementelle Routingprozess von FoG sorgt mit Hilfe dieser Daten für eine korrekte Zustellung. Anhang D.2 zeigt eine detaillierte Darstellung einer NRG-Instanz.

4.3.3 Routingmanager

Entsprechend Abschnitt 2.3.4.4 kann ein FoG-Paket die komplette Route, Teile der Route oder nur die Adresse des Ziels beinhalten. In letzterem Fall stellt die FoG-Paketweiterleitung an die lokale Routingdienstinstanz automatisch eine explizite Routinganfrage. Diese enthalten ebenfalls etwaige signalisierte Qualitätsanforderungen der sendenden Anwendung. Dadurch kann der FoG-spezifische Routingmanager für HRM sehr einfach gehalten werden, er reicht eintreffende Anfragen direkt an den Routingalgorithmus weiter. Der weitere Ablauf entspricht den Erläuterungen von Abschnitt 3.7 und übergibt das jeweilige Paket abschließend zurück an die FoG-spezifische Paketweiterleitung.

4.3.4 Routingalgorithmus

Im Gegensatz zu der in Abschnitt 3.8.3 dargestellten Auftrennung werden innerhalb der Implementierung die Routingstrategien WSPF und SWPF nicht separat ausgeführt. Stattdessen wird die lokale Routingtabelle einmalig nach den jeweils besten Einträgen pro Strategie durchsucht. Dabei wird jeder Eintrag sowohl mit den zu erfüllenden Qualitätsanforderungen als auch der für die jeweilige Strategie bisher besten Lösung verglichen.

```
GET_ROUTING_DECISION(table, dest, dr, delay)
   for i = 0 to table.length - 1
       entry = table[i]
      if matchesDestination(entry, dest)
          if providesBetterWSPF(entry, dest, dr, delay, routeWSPF)
              routeWSPF = entry
          if providesBetterSWPF(entry, dest, dr, delay, routeSWPF)
              routeSWPF = entry

   result = routeWSPF
   if routeWSPF.util >= UTIL_MAX or routeWSPF.dr - dr < DR_MIN or BE_ROUTING
      result = routeSWPF

   return result
```

Abbildung 4.7: die zentrale Funktion zur Bestimmung einer Routingentscheidung

Abbildung 4.7 zeigt den Ablauf der Routingentscheidung als Pseudocode [126]. Die verwendeten Parameter besitzen dabei folgende Bedeutung:

- *table*: die lokale Routingtabelle
- *dest*: das Ziel der Routinganfrage
- *dr*: die gewünschte Datenrate
- *delay*: die gewünschte maximale Gesamtverzögerung

Jede für das gewünschte Ziel *dest* zutreffende Route wird für beide Strategien gleichzeitig geprüft.

4.3.4.1 WSPF-basiertes Routing

Für WSPF-basiertes Routing wird die beste gefundene Route in *routeWSPF* gespeichert. Sie wird anhand folgender Priorisierung der Vergleichskriterien durch die Funktion *providesBetterWSPF* ermittelt:

1.) **Hop-Distanz:** WSPF-basiertes Routing fokussiert auf die Länge einer Route. Daher ist die Hop-Distanz in diesem Fall primäres Entscheidungskriterium.
2.) **Datenrate:** Die maximal verfügbare Datenrate entlang einer Route stellt das sekundäre Kriterium während der Iteration durch die Routingtabelle für WSPF-Routing dar. Sollte im Vergleich zur bisher ausgewählten besten Route die aktuell geprüfte eine höhere Datenrate aufweisen, wird sie als neue Lösung für WSPF-basiertes Routing gespeichert.

3.) **Datenrate zum nächsten Knoten:** Die Datenrate des Links zum nächsten Knoten wird als tertiäres Kriterium verwendet.
4.) **Verzögerung:** Die minimal zu erwartende Verzögerung entlang einer Route stellt das letzte Kriterium dar.

Diese Priorisierung entspricht einer exakten Umsetzung des in Abschnitt 3.8.3.2 aufgestellten Kostenmodells (Metrik).

4.3.4.2 SWPF-basiertes Routing

Für SWPF-basiertes Routing speichert die Implementierung die beste gefundene Route in *routeSWPF* ab. Sie wird von *providesBetterSWPF* nach folgender Priorisierung der Vergleichskriterien ermittelt:

1.) **Datenrate:** SWPF-basiertes Routing liefert die Route als Ergebnis, welche die größte noch verfügbare Datenrate zum Zielknoten bereitstellt.
2.) **Datenrate zum nächsten Knoten:** Die Datenrate des Links zum nächsten Knoten wird als sekundäres Kriterium verwendet.
3.) **Verzögerung:** Analog zu WSPF-Routing wird die Datenrate gegenüber der Verzögerung höher priorisiert, sodass die Verzögerung eine untergeordnete Gewichtung besitzt.
4.) **Hop-Distanz:** Die Länge der Route besitzt bei SWPF-Routing den geringsten Einfluss auf die Routingentscheidung. Sie muss dennoch beachtet werden, um bei gleichen QoS-Eigenschaften aufgrund von unterschiedlichen Längen die jeweils kürzere Route als Ergebnis zu verwenden.

Diese Priorisierung folgt Abschnitt 3.8.3.3 und setzt die geforderte Prioritätsverteilung zwischen den QoS-spezifischen Attributen einer Route um.

4.3.4.3 Routingentscheidung

Als Resultat der beiden parallel ablaufenden Routingberechnungen wird auf Basis eines Durchlaufes für beide Strategien die jeweils beste Lösung ermittelt. Die resultierende Routingentscheidung wird anschließend entsprechend des in Abschnitt 3.8.2.2 beschriebenen Ablaufes bestimmt. Dazu werden die gespeicherten Eigenschaften der ermittelten WSPF-Route mit den Konstanten *UTIL_MAX* und *DR_MIN* verglichen[4]. Hat diese Lösung der WSPF-Strategie nicht mehr die erforderlichen Kapazitäten, wird stattdessen das Ergebnis der SWPF-Strategie verwendet und dadurch entferntere Regionen des Netzwerks in die Übertragung der Anwendungsdaten einbezogen.

4.3.4.4 Best-Effort-Routing

Zusätzlich zu dem zuvor beschriebenen QoS-Routing bietet die Implementierung ebenfalls die Möglichkeit, dass im Netzwerk ausschließlich BE-Routing angewandt wird. In Abbildung 4.7 ist dies anhand des Vergleichs mit der globalen Konstante *BE_ROUTING* ersichtlich. Durch diese Funktionalität wird ein direkter Vergleich zwischen HRM- und BE-basiertem Routing ermöglicht. Bei Aktivierung der Funktion degradiert der Routingalgorithmus von dem in Abschnitt 3.8.2 beschriebenen Ablauf zu einem auf der WSPF-Strategie basiertem Routing. Für die innerhalb dieser Arbeit betrachteten Szenarien führt dies im Vergleich zu reinem BE-Routing zu äquivalenten Routingentscheidungen.

4.4 Programmierschnittstelle für Anwendungen

Im Allgemeinen werden die Funktionen heutiger Betriebssysteme gegenüber Anwendersoftware mit Hilfe einer sogenannten Programmierschnittstelle bereitgestellt. Sie wird auch als *Application Programming Interface* (API) bezeichnet und beinhaltet notwendige Funktionen zur Kommunikation in Netzwerken. Sowohl die Spezifikation der Schnittstelle als auch die dahinter befindliche Implementierung

[4] Innerhalb der Implementierung wurde beispielhaft für UTIL_MAX ein Wert von 95 (entspricht 95%) und für DR_MIN ein Wert von 128 (entspricht 128kbit/s) verwendet. Beide Größen müssen für eine Anwendung in realen Netzwerken an die jeweiligen Anforderungen an das gewünschte Routing angepasst werden.

können sich dabei von System zu System unterscheiden. In heutigen Implementierungen für IP wird typischerweise eine Schnittstelle zu den *Sockets* [127] eingesetzt.

4.4.1 Typische Funktionen von FoGSiEm

Innerhalb der FoGSiEm Software ist ebenfalls eine API integriert. Sie erlaubt FoG-Anwendungen die Kommunikation innerhalb des darunterliegenden FoG-Netzwerks. Hinter dieser Schnittstelle befinden sich die drei Dienste von FoG. Einer dieser Dienste ist der jeweils aktive Routingdienst. Er muss die Implementierung der notwendigen Funktionen zur Bereitstellung der API beinhalten und seine implementierungsspezifischen Eigenschaften verbergen.

Die allgemeine API von FoGSiEm zur Kommunikation im Netzwerk unterstützt neben der Festlegung des Ziels und Ereignisbehandlung auch Anforderungen für jede Verbindung. Dazu zählt insbesondere die Beschreibung von Qualitätsanforderungen für die Übertragung, welche durch die sendende Anwendung festgelegt und durch FoG an jeden beteiligten Routingdienst übermittelt werden. Der HRM-spezifische Routingdienst integriert sich dabei vollständig in diese Architektur und beinhaltet die für die API notwendigen Funktionen.

4.4.2 Erweiterungen von FoGSiEm

Neben den zuvor beschriebenen FoG-typischen Funktionen bietet die HRM-Implementierung weitere Funktionen zur Abfrage verfügbarer Netzkapazität. Die Implementierung dieser Funktionen ist Bestandteil der *HRM-Controller* Anwendung. Der Zugriff auf die genannten Funktionen geschieht in Form einer Bibliothek und stellt eine HRM-spezifische Erweiterung der API von FoG dar[5].

```
long getMaxDataRate( HRMID pDestination )
long getMinDelayAtMaxDataRate( HRMID pDestination )
long getMinDelay( HRMID pDestination )
long getMaxDataRateAtMinDelay( HRMID pDestination )
```

Abbildung 4.8: GET-Funktionen zur Ermittlung der Kapazitäten in Richtung eines gegebenen Zielknotens

Abbildung 4.8 gibt einen Überblick über alle GET-Funktionen des HRM-Routingdienstes. Sie benötigen jeweils einen Parameter, welcher die HRMID des gewünschten Zielknotens spezifiziert. Die Rückgabewerte ermittelt der Routingdienst jeweils auf Basis der lokal vorhandenen Routingtabelle. Die Funktionen vereinfachen die lokal bekannten Routingdaten insofern, dass sie keine Liste verfügbarer QoS-Eigenschaften aller bekannten Routen zurückgeben. Sie liefern stattdessen jeweils nur Maxima und Minima für die Gesamtroute zum jeweiligen Ziel. Die einzelnen Funktionen besitzen folgende Bedeutung:

- **getMaxDataRate( ):** ermittelt die maximal verfügbare Datenrate zum Ziel
- **getMinDelayAtMaxDataRate( ):** bestimmt die minimale Verzögerung entlang der Route zum Ziel, wenn die Route mit der maximal verfügbaren Datenrate verwendet wird
- **getMinDelay( ):** gibt die minimal zu erwartende Verzögerung zum Ziel zurück
- **getMaxDataRateAtMinDelay( ):** ermittelt die maximale Datenrate entlang der Route zum Ziel, wenn die Route mit der minimalen Verzögerung verwendet wird

Auf Basis dieser Funktionen sind sowohl die verfügbaren Datenraten als auch Verzögerungen für gewünschte Zielknoten bestimmbar. Eine mögliche Anwendung dafür stellt eine Videokonferenzsoftware dar. Sollte der zurückgelieferte Wert der verfügbaren Datenrate bereits unterhalb der Mindestanforderungen für den aktuell ausgewählten Videocodec liegen, kann die Anwendung automatisch auf einen alternativen Codec umschalten, welcher mit akzeptablen Qualitätseinbußen eine wesentlich geringere

[5] Die Funktionen können ebenfalls für einen IP-basierten Netzwerkstack migriert werden, da sie keine semantische Abhängigkeit zu FoG beinhalten.

Datenrate verursacht. Es ist eine entsprechende Heuristik denkbar, welche automatisch in Abhängigkeit von den aktuellen Netzkapazitäten alle Einstellungen für einen audiovisuellen Datenstrom vornimmt. Die Wiedergabequalität auf Seiten des empfangenden Konferenzteilnehmers kann dadurch verbessert werden [128].

4.5 Testanwendungen

Zur Untersuchung des Verhaltens der HRM-Implementierung unter bestimmten Lastsituationen im Netzwerk wurden die zwei Testanwendungen *QoSTestApp* und *HRMTestApp* implementiert. Die Anwendung *QoSTestApp* kann auf einem beliebigen Knoten gestartet werden. Als Konfigurationsparameter verlangt sie den Namen eines Zielknotens. Die Anwendung kann anschließend dazu verwendet werden beliebig viele Verbindungen zum Zielknoten zu erstellen und wieder zu löschen. Dabei werden an das Routing definierte Qualitätsanforderungen für die resultierende Übertragung übermittelt.

Die Anwendung *HRMTestApp* kombiniert die Instanzen der Anwendung *QoSTestApp*, welche über verschiedene Knoten verteilt sind. Dadurch können beliebige Kombinationen aus Knotenpaaren verwendet werden, um zwischen ihnen Verbindungen zu erstellen. Dies wird beispielsweise in Kapitel 6 angewandt, um einen quantitativen Vergleich der Performanz von HRM- und BE-basiertem Routing für definierte Netzwerkauslastungen zu ermitteln.

4.6 Diskussion der Implementierung

Nachfolgend werden die in Abschnitt 4.1.1 aufgestellten Anforderungen an die Implementierung aufgegriffen und jeweils erläutert, inwiefern sie innerhalb der Implementierung des Routingdienstes Beachtung finden.

4.6.1 Implementierte Funktionalitäten

Die Implementierung von Kapitel 4 umfasst alle in Kapitel 3 vorgestellten Prozesse des HRM-Konzeptes. Dazu zählen sowohl die konzipierten Protokolle der Kontrollebene als auch der in Abschnitt 3.8 vorgestellte Routingalgorithmus.

4.6.2 Dezentrale Funktionsweise

Die Implementierung verwendet die Paketsimulation von FoGSiEm als Basis, welche den einzigen Kommunikationsweg zwischen den *HRM-Controllern* auf unterschiedlichen Knoten darstellt und somit für die Signalisierung zwischen den Entitäten der Kontrollebene verwendet wird. Jegliche Synchronisationsmechanismen innerhalb der Implementierung sind dadurch vollkommen dezentral auf Basis von allgemeiner Netzwerkkommunikation umgesetzt.

4.6.3 Deterministische Ereignisverarbeitung

Im Kontext der Implementierung verteilter Signalisierungskonzepte, wie dies für HRM zutrifft, ist insbesondere die Synchronisation nebenläufiger Ereignisse eine besondere Herausforderung. Innerhalb der Implementierung wird dies durch den zentralen Thread *Processor* der Anwendung *HRM-Controller* realisiert. Die in Abschnitt 3.6.1 erläuterte Integration von TCP-Mechanismen stellt dabei zusätzlich sicher, dass Pakete eines Sendeknotens ausschließlich in chronologischer Reihenfolge am empfangenden *HRM-Controller* eintreffen. Der jeweils zugehörige *Processor* leitet diese Pakete ihrer Reihenfolge entsprechend an die jeweilige knotenlokale Entität weiter und stellt somit die chronologische Abarbeitung eintreffender Pakete sicher. Erst nach vollständiger Abarbeitung eines Pakets und der dadurch ausgelösten lokalen Ereignisses startet der *Processor* die Verarbeitung des nächsten Pakets[6].

[6] Innerhalb des Quellcodes bestehen die Namen aller Funktionen, welche ein bestimmtes Ereignis verarbeiten, aus zwei Teilen. Der erste ist das Präfix *event*, im zweiten Teil ist der Name des jeweiligen Ereignisses beschrieben. Dadurch wird die Nachvollziehbarkeit der Signalisierungen unterstützt.

4.6.4 Statistiken für Signalisierungen

Die Implementierung bietet verschiedene Möglichkeiten zur Generierung von Statistiken, welche automatisch in Dateien gespeichert werden können. Dies wird insbesondere in Kapitel 6 zur Bemessung des Datenaufkommens verwendet, welches durch die Signalisierungen der Kontrollebene verursacht wird.

4.6.5 Simulation von BE-Routing

Zusätzlich zum HRM-basierten Routing ermöglicht die Implementierung entsprechend Abschnitt 4.3.4.4 die Simulation von BE-Routing. Dadurch ist ein direkter Vergleich zwischen HRM- und BE-basiertem Routing möglich.

4.6.6 Allgemeingültigkeit

Die Implementierung erlaubt eine einfache Portierung für reine IP-basierte Netzwerkstacks. Hierfür sind die folgenden Anpassungen notwendig:

- **Kontrollebene**
  - **Nachbarschaftserkennung:** Innerhalb von FoGSiEm wird die Erkennung von direkten Nachbarn auf Basis des FoG-spezifischen Hallo-Protokolls und seiner automatisch generierten Ereignisse realisiert. Für IP-basierte Netzwerke muss dies auf Basis von Ethernet Frames umgesetzt werden. Um alle Knoten einer Broadcast-Domäne zu ermitteln, können *AnnounceNeighborNode*-Nachrichten periodisch an die Broadcast-Adresse FF:FF:FF:FF:FF:FF versandt werden. Diese müssen nachfolgend von jedem Empfänger entsprechend Abschnitt 3.3.1 beantwortet werden.
  - **Routing:** Für FoGSiEm ist es notwendig, neben Knoten-IDs ebenfalls die FoG-spezifischen Gatenummern zur Festlegung von Routen für Signalisierungspakete der Kontrollebene zu verwenden. Für eine reine IP-basierte Implementierung genügt hingegen der Einsatz von Knoten-IDs zur Beschreibung von Zwischenknoten und Ziel der Übertragung. Zusätzlich muss ein Knoten lokal eine Abbildungstabelle speichern, aus der er für eine Knoten-ID eines Nachbarknotens die zugehörige Adresse für das verwendete Protokoll von Schicht 2 auslesen kann. Die Daten dafür erhält er aus den *AnnounceNeighborNode*-Nachrichten der Kontrollebene.
- **Datenebene**
  - **Routingmanager:** Da der verwendete Routingmanager FoG-spezifisch umgesetzt ist, muss er für eine reine IP-basierte Implementierung in entsprechend angepasster Form vorliegen. IP-Pakete müssen abgefangen, die entsprechende Routinganfrage an den Routingalgorithmus gestellt und anschließend die Pakete wieder an die originäre IP-Paketweiterleitung übergeben werden.
  - **Ressourcenreservierung:** Etwaige Ressourcenreservierungen müssen bei einem IP-Netzwerkstack mit Hilfe der dort vorhandenen Funktionen implementiert sein.
  - **Routing:** Innerhalb von FoGSiEm muss die HRMID des nächsten Knotens stets auf eine FoG-spezifische Route zum jeweiligen Nachbarknoten abgebildet werden. Bei der Verwendung eines IP-Netzwerkstacks muss stattdessen die Adressierung des verwendeten Protokolls von Schicht 2 zum Einsatz kommen. Beispielsweise muss bei Einsatz von Ethernet die HRMID des nächsten Knotens auf die MAC-Adresse des jeweiligen Nachbarknotens abgebildet werden. Hierfür muss ein Knoten lokal eine Abbildungstabelle speichern. Die Daten dafür erhält er aus den *AnnounceNeighborNode*-Nachrichten der Kontrollebene. Die resultierende Tabelle kann dabei ebenfalls die jeweilige Knoten-ID des Nachbarknotens enthalten, sodass sie die zuvor beschriebenen Anpassungen für das Routing von Signalisierungsnachrichten der Kontrollebene ermöglicht.

Da die Implementierung der Anwendung *HRM-Controller*, der beiden Topologiegraphen HRG und NRG sowie des Routingalgorithmus keine FoG-spezifischen Mechanismen verwenden, können diese für IP-basierte Netzwerke unverändert zum Einsatz kommen.

4.6.7 Grafische Ausgaben zur Beobachtung

Mit den integrierten grafischen Ausgabemöglichkeiten ist eine Überwachung der Abläufe in Echtzeit möglich. In Anhang D sind dazu ausgewählte Beispiele beschrieben.

4.6.8 Model-View-Controller

Die verwendete Unterteilung der Implementierung in die zwei Plug-Ins *fog.routing.hrm* und *fog.routing.hrm.view* folgt dem Softwarekonzept „Model-View-Controller“, wodurch eine einfache Erweiterung und Wartbarkeit der Implementierung bei zukünftigen Forschungsarbeiten unterstützt wird.

4.7 Schlussfolgerungen

Innerhalb von Kapitel 4 wird der erste praktische Teil dieser Arbeit beschrieben. Er enthält die vollständige Implementierung von HRM auf Basis des Netzwerksimulators FoGSiEm. Dadurch wird die Machbarkeit einer Umsetzung des neuartigen Routingmanagements bewiesen und eine Basis für die Evaluierung von HRM anhand konkreter Netzwerkssimulationen geschaffen. Die notwendigen Signalisierungen wurden dabei entsprechend dem Konzept unabhängig von bestehenden Protokollen von Schicht 3 auf Basis einer paketbasierten Simulation umgesetzt. Dabei wurde durch den Einsatz von geeigneten Softwarekonzepten sichergestellt, dass jegliche Ereignisverarbeitung deterministisch erfolgt. Für eine Beobachtung des Netzwerkzustandes und der laufenden Signalisierungen beinhaltet die Implementierung zusätzlich entsprechende Sensorik, deren Daten zu Statistiken zusammengefasst und lokal in einer Datei abgespeichert werden können. Dadurch wird eine Bemessung des auftretenden Datenaufkommens ermöglicht, welcher aufgrund der Signalisierungen der Kontrollebene verursacht wird. Dies stellt die Grundlage für die in Kapitel 6 vorgestellte Evaluierung von HRM dar. In diesem Kontext war es zudem wichtig, dass die Implementierung ebenfalls die Simulation von BE-basiertem Routing unterstützt, sodass dadurch ein direkter Vergleich des Nutzens von HRM-basiertem Routing gegenüber BE-basiertem Routing ermöglicht wird.

Die Implementierung innerhalb der Software FoGSiEm hat gezeigt, dass sich HRM für den Einsatz in FoG-basierten Netzwerk eignet und zudem eine sinnvolle Ergänzung in Form eines eigenständigen Routingdienstes für FoG darstellt. Zusätzlich ist zu erkennen, dass die Umsetzung auch leicht für IPv4/v6-basierte Netzwerke migriert werden kann. Weitere Details dazu sind dem Abschnitt 4.6.6 zu entnehmen. Durch ihre zusätzlichen grafischen Ausgabemöglichkeiten eignet sich die Implementierung zudem für die Darstellung der HRM-Signalisierungen in Echtzeit. Einige beispielhafte Darstellungen dazu sind in Anhang D.1 zu finden. Der resultierende Quellcode der Implementierung steht der Öffentlichkeit unter [123] für weitere zukünftige Experimente und Messungen zur Verfügung.

5 Implementierung einer Testumgebung für audiovisuelle Datenströme

Der nachfolgende zweite Implementierungsteil basiert auf der Videokonferenzsoftware *Homer-Conferencing* (kurz: Homer) [11]. Sie wurde als Open-Source-Lösung für Linux, Windows und OS X unter Verwendung von vorhandenen Bibliotheken hauptsächlich durch den Autor dieser Arbeit entwickelt. Homer verwendet das *Session Initiation Protocol* (SIP) [129] zur Einrichtung von Videokonferenzen in Echtzeit und kann dadurch ebenfalls mit existierenden Videokonferenzsystemen kombiniert werden. Dadurch sind Konferenzen zwischen Homer und handelsüblichen Bildtelefonen möglich. Der Quellcode von Homer ist öffentlich zugänglich [130] und es existieren offizielle Pakete für verschiedene Linux-Distributionen.

Die Arbeit an Homer begann im Jahr 2008 im Kontext der Mitarbeit des Autors bei der Firma MetraLabs GmbH[1] im Rahmen des EU-Projektes *Companionable*[2]. Im Verlauf dieses Projektes wurde die Software dafür entwickelt, älteren Menschen die audiovisuelle Kommunikation mit Verwandten und Betreuern zu ermöglichen. Anschließend an diese Projektzeit wurde die Entwicklung im universitären Umfeld fortgesetzt, sodass als Resultat im Februar 2013 die Version 0.25 entstand. Sie bietet als erste Version den vollständigen Umfang der vom Autor geplanten Funktionalitäten. Die Software besteht zum Zeitpunkt dieser Arbeit aus rund 55000[3] Zeilen Quellcode, welche in 260 Dateien mit einer Gesamtgröße von 3,6 MB verteilt sind. Ausgewählte Teile dieser Software wurden zusätzlich in die Software *FoG-SiEm* integriert, sodass Videostreaming ebenfalls für FoG-Netzwerke verwendet werden kann. Die aus Homer und der Integration in *FoGSiEm* entstandenen Funktionen zur Generierung, Verarbeitung und Wiedergabe audiovisueller Datenströme konnten bereits zum qualitativen Vergleich des HRM-Konzeptes mit BE-Routing [131] verwendet werden. Des Weiteren wurde die Software im Rahmen des Projektes G-Lab_FoG mehrfach zur Demonstration von FoG-Netzwerken [132] [133] [134] eingesetzt. Ausgewählte Teile der Software wurden zusätzlich in den Quellcode der öffentlich verfügbaren Multimedia-Bibliotheken *FFmpeg*[4] und *libav*[5] durch den Autor dieser Arbeit integriert:

- **RTP-Paketierung/Parser für H.261**: Durch diese Erweiterung wird ein RTP-basiertes Videostreaming mit dem Videocodec H.261 ermöglicht.
- **RTP-Paketierung/Parser für HEVC**: Der Videocodec HEVC ist der direkte Nachfolger von H.264 und ist ebenfalls unter der Bezeichnung H.265 zu finden. Er stellt für heutiges Videostreaming die aktuellste Version seiner Codecfamilie dar. Durch die durchgeführten Erweiterungen steht ein RTP-basiertes Videostreaming auf Basis von HEVC für beide Bibliotheken zur Verfügung, sodass erstmalig eine synchrone Wiedergabe von Bild und Ton einer entfernten Quelle unter Verwendung des aktuellen Videocodecs HEVC ermöglicht wird. Die integrierten Funktionen werden heute ebenfalls durch den bekannten Mediaplayer *VLC*[6] eingesetzt, sodass auf diesem Wege auch dessen Anwender von den durchgeführten Arbeiten profitieren.
- **Übertragung auf Basis von UDP-Lite**: Das Protokoll UDP-Lite eignet sich insbesondere zum Übertragen von audiovisuellen Daten im Kontext alternativer Bitfehler-tolerierenden Implementierungen für Schicht 2 des OSI-Modells. Dadurch treten bei der Wiedergabe auf Empfängerseite weniger Bildausfälle auf, stattdessen führen Bitfehler häufig nur zu vereinzelten fehlerhaften Bildbereichen.

[1] http://metralabs.com/
[2] http://www.companionable.net/
[3] Dieser Wert wurde mit der Software *cloc* ermittelt, sie ist unter http://cloc.sourceforge.net/ erhältlich.
[4] http://www.ffmpeg.org/
[5] https://libav.org/
[6] http://www.videolan.org/

Im Gegensatz zu *Ekiga*[7] und *Linphone*[8], den beiden bekanntesten alternativen Open-Source-Videokonferenzlösungen, unterstützt Homer neben UDP und TCP ebenfalls die Transportprotokolle UDP-Lite sowie SCTP. Im Zuge von Evaluierungsmessungen mit dem Protokoll UDP-Lite wurde durch den Autor eine fehlerhafte Implementierung im Quellcode des Linuxkernels aufgedeckt, welche in Zusammenarbeit mit dem ursprünglichen Autor der entsprechenden Teile des Linuxquellcodes korrigiert wurde[9].

Im Allgemeinen ist Homer als Testbett und Studienobjekt für Multimedia-Streaming zu verstehen, dessen Möglichkeiten in [135] der Öffentlichkeit vorgestellt wurde. Die Funktionen der Software können zum qualitativen Vergleich zwischen verschiedenen Routingansätzen an realen Beispielen für audiovisuelle Datenströme verwendet werden. Dafür bietet die Oberfläche von Homer grafische Ausgabemöglichkeiten zur Beobachtung und Bemessung der eingesetzten Datenströme. Der nachfolgende Abschnitt 5.1 stellt die implementierte Softwarearchitektur im Überblick vor. Dabei wird zuerst auf die ursprünglichen Anforderungen an die Software eingegangen und anschließend wird die daraus resultierende modulare Struktur der Software vorgestellt. Auf die wichtigsten Softwaremodule wird anschließend in den darauffolgenden Abschnitten detaillierter eingegangen. Dabei sind im wissenschaftlichen Kontext insbesondere die Ausführungen aus Abschnitt 5.5 wichtig: Homer verwendet die Programmierschnittstelle *GAPI* [136]. Dadurch können die in Homer erzeugten Datenströme mit Hilfe von verschiedenster Implementierungen eines Netzwerkstacks und den damit verbundenen Transportprotokollen übertragen werden. Für Homer steht sowohl eine Anbindung an den IP-Netzwerkstack (Abschnitt 5.5.1) als auch für den FoG-Netzwerkstack (Abschnitt 5.5.2) zur Verfügung. Letzterer ermöglicht eine Kombination von Homer mit dem in *FoGSiEm* laufenden HRM-Routingdienst, sodass eine direkte Nutzung von HRM in einer realen Anwendung ermöglicht wird. Das Kapitel endet inhaltlich in Abschnitts 5.8 mit einem Vergleich der in Abschnitt 5.1.1 aufgeführten Anforderungen und der vorliegenden Umsetzung in Homer.

5.1 Softwarearchitektur

Bevor die Gesamtarchitektur von Homer erläutert werden kann, muss ein Überblick über die im Vorfeld der Implementierung aufgestellten Anforderungen gegeben werden.

5.1.1 Anforderungen

Im Kontext der MetraLabs GmbH bestand das vordergründige Ziel in der Entwicklung in einer unabhängigen, eigenständigen Videokonferenzlösung. Im Zuge der Fortsetzung der Arbeit an der TU Ilmenau rückten die Wünsche nach Erweiterbarkeit und Integration neuer Netzwerktechnologien in den Vordergrund. Neben den typischen Videokonferenzfunktionalitäten waren zusammenfassend folgende Kernfunktionen für die Untersuchung von Übertragungen gefordert:

- **Codec- und Qualitätseinstellungen:** Es müssen verschiedene Video- und Audiocodecs unterstützt werden. Grafische Dialoge sollen die Möglichkeit bieten, die Verarbeitung audiovisueller Datenströme zu parametrisieren. Auf dieser Basis sollen verschiedenartige Charakteristiken der resultierenden Netzwerkpakete hervorgerufen werden können. Dazu zählen insbesondere die resultierende Datenrate sowie die Verteilung von Paketgrößen.
- **Übertragungsanforderungen pro Datenstrom:** Für jeden Datenstrom muss es möglich sein, explizite Anforderungen an die Übertragung festzulegen, sodass dadurch Routingentscheidungen im Netzwerk beeinflusst werden. Dies ist mit alternativen Open-Source-Softwarelösungen, wie *Ekiga* oder *Linphone*, nicht möglich.

[7] http://www.ekiga.org/
[8] http://www.linphone.org/
[9] https://www.kernel.org/pub/linux/kernel/v3.x/ChangeLog-3.1

- **Austauschbarer Netzwerkstack:** Die Software muss es ermöglichen, den aktuellen IP-basierten Netzwerkstack inklusive seiner Vielfalt an Protokollen zu verwenden. Dazu zählen, neben den klassischen Transportprotokollen TCP und UDP, auch Alternativen wie UDP-Lite und SCTP. Des Weiteren soll die Software zukünftige Netzwerkstacks unterstützen. Dazu zählt insbesondere die Nutzung von FoG-basierten Netzwerken, sodass ein direkter Vergleich zwischen heutigem BE- und zukünftigem HRM-basiertem Routing anhand der subjektiv empfundenen Wiedergabequalität auf Empfängerseite durchgeführt werden kann.
- **Beobachtung und Messung von Datenströmen:** Die Software soll grafische Dialoge zur Beobachtung und Messung einzelner Datenströme beinhalten. Dies soll zur Charakterisierung von audiovisuellen Daten in Echtzeit und der Ableitung von Heuristiken zur Bestimmung erforderlicher Datenraten in Abhängigkeit von den Codec- und Qualitätseinstellungen einsetzbar sein.

Zusätzlich wurden weitere Anforderungen aufgestellt, welche den Einsatz der Anwendung insbesondere für wissenschaftliche Untersuchungen unterstützen:

- **Dezentralisierte Arbeitsweise:** Alle Funktionen sollen ohne zusätzliche Infrastruktur verwendbar sein. Dafür muss die Software jegliche Managementsignalisierungen auch ohne einen zentralen Ankerpunkt im Netzwerk ausführen können.
- **Eigenständigkeit der Lösung:** Es soll möglich sein, die Software ohne Fremdsoftware verwenden zu können.
- **Systemunabhängigkeit:** Die Software soll sowohl für Linux, Windows als auch OS X einsetzbar sein. Dadurch soll eine hohe Flexibilität ihrer Verwendung untermauert werden.
- **Allgemeingültigkeit:** Unabhängig vom jeweils verwendeten Netzwerkstack soll die Software die Übertragung von audiovisuellen Datenströmen ermöglichen, sodass der direkte Vergleich zwischen verschiedenen Implementierungen von Netzwerkstacks durchgeführt werden kann.

Nachdem die ursprünglichen Anforderungen an die Architektur definiert sind, kann die daraus resultierende Architektur im nächsten Abschnitt beschrieben werden.

5.1.2 Softwaremodule

Aus Sicht des Nutzers stellt sich Homer als grafische Anwendung für Videokonferenzen dar. Hinter der grafischen Oberfläche werden sechs zusätzliche Bibliotheken verwendet, sodass sich die Software über sieben Softwaremodule erstreckt. Dabei sind die grafischen Ausgaben von allen im Hintergrund genutzten Funktionen für Konferenzmanagement, Audio-/Videoverarbeitung und Netzwerkkommunikation separiert.

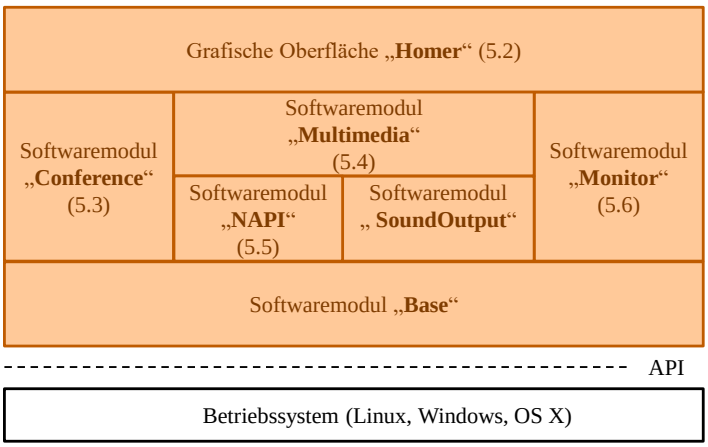

Abbildung 5.1: Softwarearchitektur von Homer-Conferencing

Abbildung 5.1 zeigt die Unterteilung von Homer in Programm und Bibliotheken. In Klammern ist dabei der jeweilige Abschnitt aufgeführt, worin weitere Details gegeben werden. Die zwischen den Softwaremodulen existierenden hauptsächlichen funktionalen Abhängigkeiten werden durch die vertikale Anordnung in Abbildung 5.1 ausgedrückt: Jedes Softwaremodul benötigt dabei die Funktionen von den darunter liegenden Modulen.

Modul	Funktion
Homer	• Interaktion mit dem Nutzer auf Basis einer grafischen Benutzeroberfläche • Wiedergabe von Videoströmen
Conference	• Konferenzverwaltung - Parametrisierung von audiovisuellen Datenströme
Multimedia	• Hardwarezugriff auf Audio/Videoquellen • Audiovisuelle Verarbeitung - (De-)Kodierung von Audio-/Videoströmen - RTP-basierte Paketierung von Datenströmen • Hardwarezugriff zur Audioausgabe unter Linux und Windows
SoundOutput	• Steuerung des Hardwarezugriffs für Audiowiedergabe speziell für OS X
NAPI („Network Application Programming Interface“)	• Anbindung verschiedener Netzwerkstacks - IP-Netzwerkstack auf Basis von Sockets - FoG-Netzwerkstack • Multiplexing zwischen verschiedenen Netzwerkstacks pro Datenstrom
Monitor	• Überwachung von Threads • Überwachung von Datenströmen - Generierung von Statistiken
Base	• Abstraktion spezifischer Betriebssystemfunktionen von Linux, Windows und OSX

Tabelle 5.1: Softwaremodule von Homer Conferencing

Tabelle 5.1 gibt einen Überblick über die Funktionen der einzelnen Softwaremodule von Homer. Für die Systemunabhängigkeit ist die Bibliothek *Base* zuständig. Sie implementiert eine allgemeingültige Schnittstelle für die darüber befindlichen Softwaremodule. Hinter dieser abstrahiert sie die Unterschiede zwischen den drei möglichen Betriebssystemen: Linux, Windows und OS X.

5.2 Grafische Oberfläche von Homer

Für die grafischen Fenster und Dialoge nutzt die Implementierung die plattformübergreifende Bibliothek *Qt* in der aktuellen Version 5.5 der Firma *Digia Plc* aus Finnland. Die Bibliothek steht als Open-Source-Lösung zur Verfügung und prägt das Aussehen von Homer-Conferencing für Linux, Windows und OS X. Die Implementierung von Homer wählt die jeweils notwendigen Fensterelemente aus und parametrisiert ihr Aussehen. Des Weiteren beinhaltet der Quellcode die notwendige Logik zur Reaktion auf Nutzereingaben sowie zur Steuerung der Sichtbarkeit einzelner Elemente in Abhängigkeit vom Status der Software.

5.3 Softwaremodul *Conference*

Für die Konferenzverwaltung wurde das *Session Initiation Protocol* (SIP) [129] gewählt, was eine einfach einsetzbare Lösung zur Konferenzverwaltung darstellt. Als bekannteste Alternative existiert das Protokoll H.323 [137]. Im Vergleich zu SIP verlangt der Einsatz dieser Lösung eine signifikant höhere Implementierungskomplexität. Neben dem eigentlichen H.323 Protokoll müssen häufig auch seine vielfältigen Unterprotokolle H.225, H.245, H.450 und H.235 in eine Implementierung einbezogen werden.

Eine universelle Bibliothek zur Integration dieser Protokolle stand zum Startzeitpunkt der Implementierung nicht als Open-Source-Lösung zur Verfügung[10]. Stattdessen wurde SIP auf Basis der Open-Source-Bibliothek *sofia-sip* [138] für das Softwaremodul *Conference* favorisiert. Die bei SIP genutzten Signalisierungsnachrichten setzen ein Request-Response Protokoll um. Es ist ähnlich anderen XML-basierenden Dienstprotokollen [139] aufgebaut. Als Resultat dieses Ansatzes besteht jede Signalisierungsnachricht von SIP aus direkt lesbaren Elementen in textähnlicher Form. Dies ist vorteilhaft für das Nachvollziehen der Signalisierungen des Konferenzmanagements und stellt einen weiteren Vorteil gegenüber einer Alternativlösung auf Basis von H.323 dar.

5.3.1 Verwaltung von Konferenzen

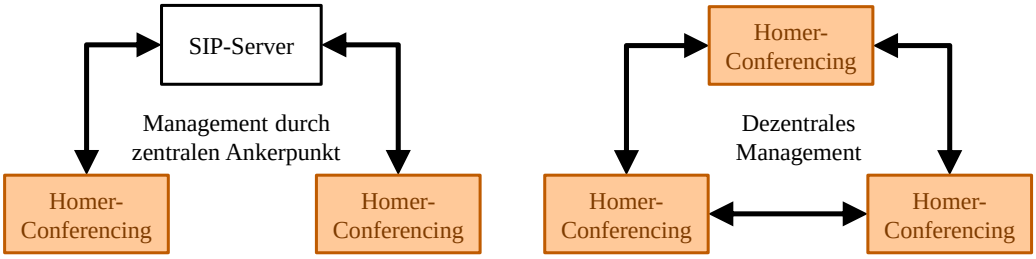

Abbildung 5.2: Szenarien des Konferenzmanagements

Auf Basis von SIP unterstützt Homer sowohl einen Servermodus als auch ein dezentrales Management für Videokonferenzen. Beide sind in Abbildung 5.2 zu sehen. Im ersten Fall auf der linken Seite kommuniziert jede Instanz von Homer mit einem zentralen SIP-Server. Dieser kann beispielsweise durch die Software *Asterisk*[11] bereitgestellt werden. Dadurch ist eine Lokalisierung möglich, sodass jeder Konferenzpartner über seine allgemein bekannte ID unabhängig seiner Position im Netzwerk gefunden und zur Konferenz eingeladen werden kann. Im zweiten dargestellten Fall kommunizieren alle Instanzen von Homer ausschließlich direkt miteinander und übernehmen eigenständig alle Managementaufgaben zur Verwaltung von Konferenzen. Eine automatische Lokalisierung ist in diesem Fall nicht durch Standardmechanismen des SIP-Protokolls gegeben und muss über eine externe Lösung realisiert werden. Unabhängig von der eingesetzten Managementstruktur können alle Video- und Audioströme direkt zwischen den Konferenzteilnehmern übertragen werden. Diese Datenströme sind in Abbildung 5.2 nicht explizit aufgeführt.

5.3.2 Beschreibung der audiovisuellen Datenströme

Für jeden Konferenzteilnehmer müssen die jeweils lokal gewählten Parameter für die Übertragung audiovisueller Daten zwischen mit den anderen Konferenzteilnehmern ausgetauscht werden, sodass auf der jeweiligen Empfängerseite eine korrekte Wiedergabe erfolgen kann. Zu den Parametern zählen der pro Teilnehmer eingesetzte Audio- und Videocodec. Zusätzlich sind Einstellungen zur Bildauflösung und verwendeter Samplerate für Video bzw. Audio möglich. Für die Aushandlung der Parameter wurde das *Session Description Protocol* (SDP) [140] gewählt. Es ist ein Unterprotokoll von SIP und verwendet ebenfalls XML-artige Signalisierungen. Das Protokoll wurde, ähnlich SIP, auf Basis von *sofia-sip* in das Softwaremodul *Conference* integriert. Die Auswahl lokaler Parameter für audiovisuelle Datenströme von Konferenzen erfolgt dabei jedoch auf Basis eines eigens dafür entwickelten grafischen Dialogs. Weitere Details dazu gibt Anhang E.1.

[10] Heute existiert mit dem *Inter-Asterisk eXchange Version 2* (IAX) eine weitere Alternative. Im Gegensatz zu SIP überträgt IAX jedoch alle Signalisierungen zusammen mit den Nutzdaten. Statt der bei SIP üblichen separaten Ports für Signalisierungen und audiovisuellen Daten verwendet IAX einen einzigen UDP-Port für alle Daten. Dieses Verhalten erschwert eine Messung einzelner Datenströme.
[11] http://www.asterisk.org/

5.4 Softwaremodul *Multimedia*

Homer bietet vielfältige Möglichkeiten der Audio- und Videoverarbeitung. Durch sogenannte *Mediamodule* werden dabei einzelne Schritte der jeweiligen Verarbeitungskette umgesetzt. Es gibt drei Typen:

- **Mediaquellen:** Sie dienen zur Gewinnung von audiovisuellen Daten. Eine Mediaquelle stellt den Start einer Verarbeitungskette dar. Dabei kann sie ihre Daten von lokal angeschlossener Hardware, einer lokalen Multimediadatei oder einem eintreffenden Netzwerkstrom beziehen. Die Implementierung verwendet dabei Funktionen der Multimediabibliotheken FFmpeg und *portaudio*[12].
- **Mediamuxer:** Ein Muxer dient dazu, die Daten einer ausgewählten Mediaquelle entgegen zu nehmen, sie entsprechend der gewählten Einstellungen neu zu kodieren und an eine oder mehrere Mediasenken weiterzugeben[13]. Es können mehrere Muxer in einer Verarbeitungskette kombiniert werden. Die bereitgestellten Funktionen der Bibliothek zur (De-)Kodierung von Video- und Audioströmen basiert dabei insbesondere auf Funktionen von FFmpeg. Weitere Details zur Video- und Audioverarbeitung sind in den Abschnitten 5.4.1 bzw. 5.4.2 zu finden.
- **Mediasenken:** Eine Mediasenke stellt den Endknoten einer Verarbeitungskette von Video-oder Audiodaten dar. Die audiovisuellen Daten können, neben einem ausgehenden Netzwerkstrom oder einer lokalen Datei, ebenfalls an lokale Hardware zur Wiedergabe geschickt werden. Für die Audioausgabe unter Linux und Windows[14] verwendet die Implementierung Funktionen der Bibliothek *portaudio*. Des Weiteren ist eine Ablage im lokalen Hauptspeicher möglich, was insbesondere für die in Abschnitt 5.7 beschriebene Integration in FoGSiEm notwendig ist.

Eine vollständige Verarbeitungskette besteht aus mindestens einer Mediaquelle, einem Muxer sowie einer Mediasenke. Ein Programmierbeispiel einer solchen Kette ist in Anhang E.5 zu finden.

5.4.1 Videoverarbeitung

Die integrierte Videoverarbeitung unterstützt die Videocodecs MPEG 1/2/4, H.261/3/3+/4 sowie den aktuellen Videocodec HEVC (H.265). Dadurch sind die bekanntesten Codecs aus dem Umfeld von SIP-basierten Konferenzlösungen durch Homer abgedeckt. Mit Hilfe der grafischen Dialoge kann einer dieser Codecs ausgewählt werden. Anhang E.1 zeigt die verfügbaren zusätzlichen Konfigurationsmöglichkeiten anhand von Beispielbildern der Anwendung.

[12] http://www.portaudio.com/

[13] Dieser Prozess wird „muxing“ genannt und im Allgemeinen von einem sogenannten „Muxer“ durchgeführt. Weitere Details sind unter https://www.ffmpeg.org/doxygen/trunk/group__lavf__encoding.html zu finden.

[14] Das Softwaremodul *SoundOutput* stellt mit Hilfe der externen Bibliotheken *SDL*, *SDL_sound* und *SDL_mixer* allgemeine Funktionen zur Audiowiedergabe zur Verfügung. Diese werden ausschließlich unter OS X eingesetzt. Aufgrund ihrer geringen Relevanz für diese Arbeit werden sie nicht detaillierter erläutert.

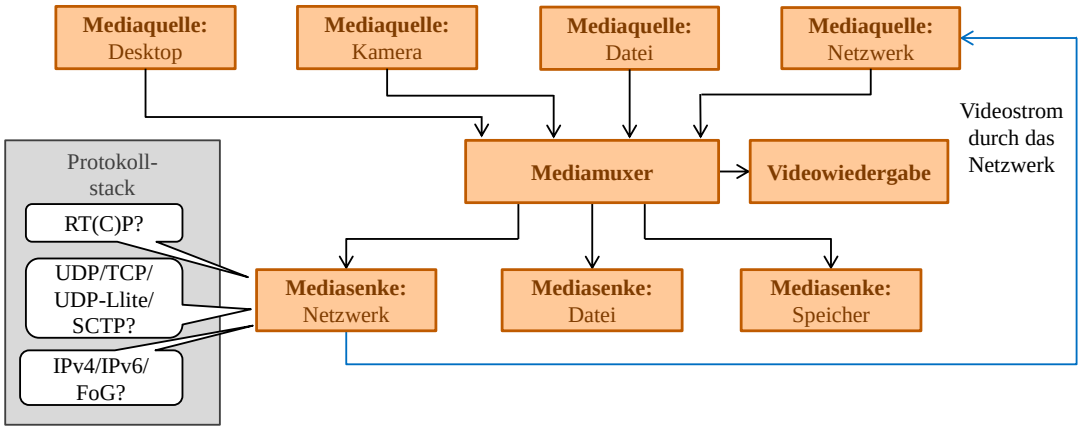

Abbildung 5.3: Datenflüsse der Videoverarbeitung

Abbildung 5.3 zeigt alle Datenflüsse, welche im Softwaremodul *Multimedia* zur Videoverarbeitung unterstützt werden. Die Videowiedergabe ermöglicht es, den ausgehenden Datenstrom eines Mediamuxers abzugreifen und innerhalb der grafischen Oberfläche auszugeben. Der mögliche Netzwerkstrom zwischen einer Mediasenke zu einer Mediaquelle einer anderen Instanz von Homer ist mit einem blauen Pfeil gekennzeichnet. Für eine Konferenz sind zwischen zwei Teilnehmern jeweils zwei dieser Datenflüsse notwendig, um bidirektional eine Videoübertragung zu realisieren. In der grauen Box in der linken unteren Ecke der Abbildung ist ein Überblick über die verfügbaren Protokolle zur Netzwerkübertragung zu sehen. Für jeden Datenstrom können das *Realtime Transport Protocol* (RTP) [141] sowie sein Unterprotokoll *RTP Control Protocol* (RTCP) [141] zu- oder abgeschaltet werden. Weitere Details dazu sind in Abschnitt 5.4.3 gegeben. Des Weiteren kann jede Netzwerkübertragung mit Hilfe von verschiedenen Netzwerk- und Transportprotokolle ausgeführt werden. Abschnitt 5.5 geht darauf detaillierter ein.

5.4.2 Audioverarbeitung

Die implementierte Audioverarbeitung in Homer verwendet die Audiocodecs G.711, G.722, PCM16 sowie MP3. Sie stellen die Menge häufig genutzter Codecs im Kontext SIP-basierter Konferenzlösungen dar. Zwischen ihnen kann mit Hilfe von grafischen Dialogen gewechselt werden. Die Einstellmöglichkeiten sind in Anhang E.1 aufgeführt.

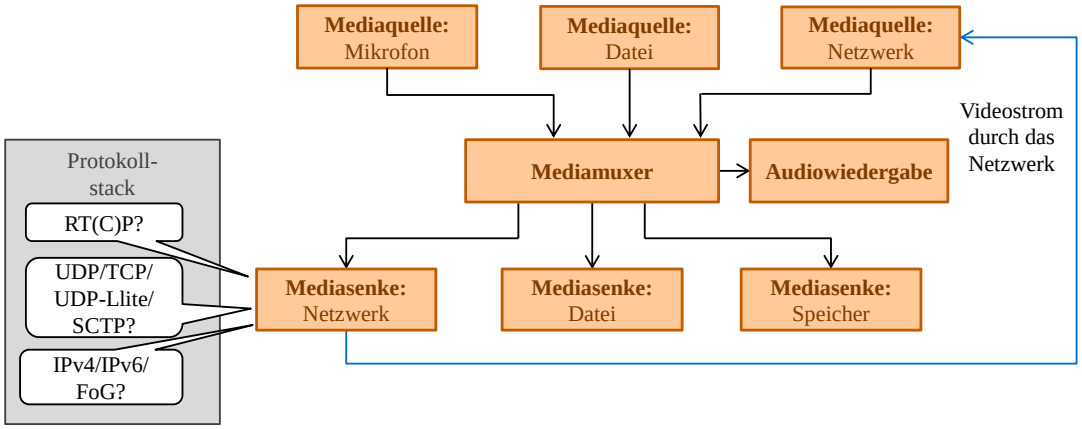

Abbildung 5.4: Datenflüsse der Audioverarbeitung

Abbildung 5.4 zeigt die vorhandenen Mediamodule zur Audioverarbeitung sowie die zwischen ihnen möglichen Datenströme. Die resultierende Struktur unterscheidet sich zur Videoverarbeitung nur minimal. Es wurde im Softwaremodul *Multimedia* zusätzlich eine Möglichkeit zur direkten Audioausgabe integriert, welche speziell für Videokonferenzen für eine Echtzeitwiedergabe notwendig ist.

5.4.3 Synchronisation zwischen Bild und Ton

Das HRM-basierte Routing leitet audiovisuelle Datenströme anhand ihrer jeweiligen Anforderungen an Datenrate und Verzögerung durch das Netzwerk. Da sich die Anforderungen zwischen Video- und Audiodaten typischerweise bezüglich der geforderten Datenrate unterscheiden, können die resultierenden Routingentscheidungen schnell zu unterschiedlichen Routen zwischen den Strömen führen. Dadurch können sich die Laufzeiten bei der Übertragung durch das Netzwerk sehr schnell unterscheiden. Zudem ist es denkbar, dass entlang einer Route die resultierende Gesamtverzögerung variiert, sodass die Ankunftszeiten von Paketen nicht mehr äquidistant ausfallen – dies wird als *Jitter*[15] bezeichnet.

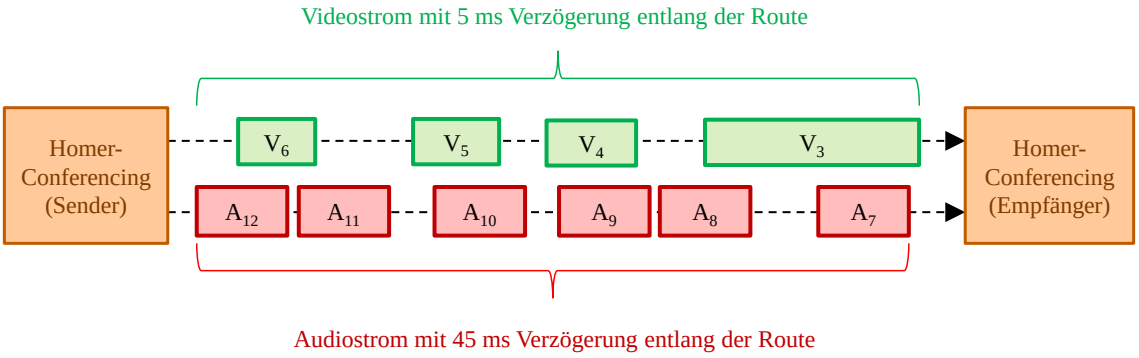

Abbildung 5.5: Gleichzeitige Übertragung von Video und Audio eines Teilnehmers

Die Abbildung 5.5 zeigt ein Beispiel eines gleichzeitigen Empfangs von Video- und Audiopaketen der gleichen Quelle. Es ist zu erkennen, dass durch Jitter die Abstände zwischen den Paketen eines Datenstroms sowie die zeitliche Differenz zwischen den Daten mehrerer Ströme variieren. Speziell bei Videokonferenzen ergeben sich daraus neue Herausforderungen für die Software. Sowohl Audio- als auch Videodaten der gleichen Quelle müssen in Echtzeit mit kontinuierlicher Geschwindigkeit und synchron zueinander auf Empfängerseite wiedergegeben werden.

5.4.3.1 Real-time Transport Protocol

Zur Studie von Möglichkeiten zur Kompensation der zuvor genannten Effekte wurde das *Realtime Transport Protocol* (RTP) in Homer verwendet. Es wurde eigens für die Übertragung von audiovisuellen Datenströmen entwickelt. Bei seiner Anwendung wird jeder Datenblock eines Video-/Audiostroms auf Senderseite in einzelne Pakete aufgeteilt, wobei im jeweiligen Paketkopf RTP-spezifische Metadaten gespeichert werden. Sie enthalten sowohl einen Zeitstempel als auch eine Sequenznummer, wobei ersterer zur Ermittlung der korrekten relativen Wiedergabezeit in Bezug auf die Startzeit einsetzbar ist. Dadurch wird eine konstante Wiedergabe des jeweiligen Datenstroms ermöglicht. Die Sequenznummern dienen wiederum, ähnlich den TCP-Sequenznummern, auf Empfängerseite zur Wiederherstellung der ursprünglichen Paketordnung. Des Weiteren können sie zur Erkennung von Paketverlusten eingesetzt werden.

Die Implementierung verwendet für die Integration der beschriebenen Mechanismen auf Senderseite die in FFmpeg existierenden Funktionen zur RTP-Paketierung. Für das Verarbeiten (Parsen) von eintreffenden RTP-Pakete auf Empfängerseite wird hingegen eine eigens dafür entwickelte Implementierung eingesetzt. Dadurch sind die RTP-Sequenznummern eintreffender Pakete innerhalb von Homer bekannt,

[15] Ein überlasteter Router kann durch Pufferung von Paketen *Jitter* in Abhängigkeit seiner Auslastung verursachen. Ähnliches gilt für Firewalls und der durch die Filterung von eintreffenden Paketen verursachten Verzögerung. Des Weiteren wird in Ethernet-basierten Netzwerken die *binary exponential backoff* Zeit zur expliziten Verzögerung von Paketen verwendet. Dadurch werden permanente Kollisionen bei der Verwendung des Übertragungsmediums vermieden und zugleich zusätzliche, variierende Verzögerungen verursacht.

sodass sie zur Beobachtung und Messung eingehender Datenströme einbezogen werden können. Nähere Details sind dazu in Abschnitt 5.6 zu finden.

5.4.3.2 RTP Control Protocol

Wie zuvor erläutert ist die Kompensation von unterschiedlichen Übertragungsverzögerungen insbesondere im Kontext von HRM interessant. In [142] wird dies bei der gleichzeitigen Übertragung von Bild und Ton als „Lippensynchronität" bezeichnet. Die Differenz zwischen Video- und Audiowiedergabe wird mit einem Maximalwert von 100 ms als akzeptabel angegeben. Zur Einhaltung dieses Wertes wird das *RTP Control Protocol* (RTCP) in Homer verwendet. Als Unterprotokoll von RTP dient es der Übertragung zusätzlicher Metadaten. Dazu zählen der absolute Zeitstempel des Senders, welche der relativen Wiedergabezeit zugeordnet werden können. Mit seiner Hilfe kann auf Empfängerseite für einen empfangenen Datenstrom auf die Zeitdifferenz zu einem anderen Datenstrom der gleichen Quelle zurückgeschlossen werden. Durch entsprechende zeitliche Verschiebung der lokal gepufferten Audio- bzw. Videodaten kann somit die gleichzeitige Wiedergabe beider Ströme kontinuierlich synchron gehalten werden.

5.5 Softwaremodul *NAPI*

Das Modul *Network-API* (NAPI) realisiert das Konzept der Programmierschnittstelle *G-Lab API* (GAPI) [136], worüber die Anwendung Zugang zu den Kommunikationsfunktionen des jeweiligen Netzwerkstacks erhält. Das Konzept der GAPI wurde im Kontext des Projektes *G-Lab* [143] entwickelt. Die Schnittstelle ist so ausgelegt, dass sich verschiedene Implementierungen unterschiedlicher Netzwerkstacks dahinter befinden können. Ihre Nutzung ist im Kontext von HRM interessant, da mit ihrer Hilfe gewünschte Eigenschaften an die Übertragung der Anwendungsdaten beschrieben werden können. Der aktive Netzwerkstack erhält diese als Anforderungen an die internen Funktionen, sodass sie durch die im Stack umgesetzten Protokolle beachtet werden können. Diese entscheiden, welche Anforderungen knotenlokal verarbeitet werden und welche den Routern im Netzwerk signalisiert werden. Die Anwendungsanforderungen unterteilen sich in zwei Klassen:

- **Funktionale Anforderungen:** Als geforderte Funktion kann beispielsweise die automatische Neuübertragung von fehlerhaften Paketen zur verlustlosen Übertragung gewählt werden. Im heutigen IP-basierten Netzwerkstack wird dies beispielsweise durch die Mechanismen von TCP oder SCTP realisiert werden.
- **Nichtfunktionale Anforderungen:** Zu diesen Eigenschaften zählen geforderte QoS-Eigenschaften, welche entlang der gesamten Route eingehalten werden sollen. Im Kontext von HRM sind dies Datenrate und Verzögerung.

Mit Hilfe der grafischen Dialoge von Homer können sowohl die Anforderungen an die Übertragung als auch der zur Übertragung zu nutzende Netzwerkstack für einen ausgehenden audiovisuellen Datenstrom gewählt werden. Beispielhaft ist dies in Anhang E.2 gezeigt. In Homer steht eine Implementierung der GAPI für den heutigen IP-Netzwerkstack zur Verfügung. Der nachfolgende Abschnitt 5.5.1 beschreibt dies näher. Alternativ dazu existiert eine Anbindung an FoG-basierte Netzwerke, welche in Abschnitt 5.5.2 näher beschrieben ist.

5.5.1 Der IP-Netzwerkstack

Für heutige IP-basierte Netzwerke nutzt die Implementierung die Socket-Schnittstelle. Zur Verfügung stehen dabei sowohl Version 4 als auch Version 6 von IP, die Auswahl der IP-Version geschieht automatisch auf Basis der gegebenen Zielbeschreibung der Anwendung.

5.5.1.1 Auswahl des Transportprotokolls

Für die Auswahl des Transportprotokolls wird eine Logik eingesetzt, welche auf Basis der geforderten funktionalen Anforderungen das Transportprotokoll wählt. Das dabei verwendete Regelwerk besteht aus disjunkten Bedingungen für jedes Protokoll. Nachfolgend sind diese in konjunktiver Normalform [144] aufgeführt:

$$\begin{aligned}
\text{TCP} &= (\text{Ordered} \vee \text{LossLess}) \wedge \text{Stream} \wedge \overline{\text{BitErrors}} \\
\text{UDP} &= \overline{\text{Ordered}} \wedge \overline{\text{LossLess}} \wedge \overline{\text{Stream}} \wedge \overline{\text{BitErrors}} \\
\text{UDP-Lite} &= \overline{\text{Ordered}} \wedge \overline{\text{LossLess}} \wedge \overline{\text{Stream}} \wedge \text{BitErrors}
\end{aligned}$$

Die verwendeten Anforderungen besitzen folgende Bedeutung:

- **Ordered:** Die Pakete sind in der gesendeten Reihenfolge dem Empfänger zuzustellen.
- **LossLess:** Die Übertragung soll verlustlos erfolgen.
- **Stream:** Die Daten des Senders werden als Strom übertragen. Durch Aneinanderreihung der Daten empfangener Pakete gewinnt der Empfänger den ursprünglichen Datenstrom zurück.
- **BitErrors:** Der Empfänger akzeptiert auch ausgewählte Pakete mit Bitfehlern. Dies wird insbesondere durch das UDP-Lite Protokoll umgesetzt.

Sollte keine der oben genannten Regeln zutreffen, wird innerhalb der Implementierung UDP als Standardauswahl angenommen.

5.5.1.2 Festlegung von Qualitätsanforderungen

Neben den funktionalen Übertragungsanforderungen unterstützt die IP-spezifische Implementierung der GAPI ebenfalls die Übermittlung von Qualitätsanforderungen:

- **LimitDataRate:** Die Datenrate wird festgelegt, welche die Anwendung für den jeweiligen Datenstrom maximal benötigt.
- **LimitDelay:** Die maximal erlaubte Gesamtverzögerung für die Übertragung bis zum Empfänger wird festgelegt.

Innerhalb der GAPI-Implementierung müssen die erhaltenen Qualitätsanforderungen wiederum an den Netzwerkstack übermittelt werden.

```
bool setQoS(int pID, int pDataRate, int pDelay)
```

Abbildung 5.6: Funktion zur Festlegung der geforderten QoS-Eigenschaften via Socket-API

Da die Standardfunktionen der API für Sockets keine Festlegung von Qualitätsanforderungen unterstützt, ist eine Erweiterung um die Funktion *setQoS* aus Abbildung 5.6 notwendig. Durch sie können einem Socket zusätzlich Qualitätsanforderungen über die folgenden Parameter zugeordnet werden:

- **pID:** Mit Hilfe dieser lokal eindeutigen ID[16] wird die Instanz eines Sockets ausgewählt.
- **pDataRate:** Die notwendige Datenrate wird in kbit/s festgelegt.
- **pDelay:** Die maximal erlaubte Verzögerung wird in Millisekunden festgelegt.

Im Kontext von Homer werden die geforderten Werte für Datenrate und Verzögerung für Signalisierungen entsprechend des *DiffServ*-Modells verwendet, sodass sie an alle Router des Netzwerks für jedes übertragene Paket signalisiert werden. Weitere Details dazu sind in Anhang E.4 zu finden.

[16] Sie wird auch als *file descriptor handle* bezeichnet.

Alternativ ist es für zukünftige Untersuchungen auch möglich, die Qualitätsanforderungen entsprechend des *IntServ*-Modells für eine explizite Pfadreservierung mit Hilfe von RSVP auf Basis von IP einzusetzen. Der Rückgabewert der Funktion *setQoS* kann in dem Fall Auskunft über Erfolg oder Misserfolg der Reservierung wiedergeben.

5.5.2 Der FoG-Netzwerkstack

Neben der Unterstützung für herkömmliche IP-Netzwerke beinhaltet Homer ebenfalls eine GAPI-Implementierung zur Anbindung an die Software FoGSiEm. Sie entstand in Zusammenarbeit [132] mit dem Institut für experimentelle Mathematik der Universität Duisburg-Essen. Durch die Anbindung können beliebige Anwendungsdaten durch FoG-basierte Netzwerke unter Beachtung von Anwendungsanforderungen übertragen werden.

Zur Anbindung an FoGSiEm werden die Möglichkeiten des *Stream Control Transmission Protocols* (SCTP) [20] ausgenutzt. Aufrufe der GAPI-Schnittstelle werden dabei zu Aufrufen der Bibliothek *libsctp*[17] abgebildet, welche eine SCTP-spezifische Erweiterung der Sockets darstellt. SCTP unterstützt pro Assoziation[18] mehrere logische Datenströme. Sie sind für eine Anwendungsinstanz sichtbar und können explizit adressiert werden. Diese Möglichkeit ist ein besonderes Merkmal von SCTP und wird im Kontext der Anbindung an den FoG-Netzwerkstack für die Unterscheidung zwischen den Daten und den Übertragungsanforderungen einer Anwendung verwendet. Die Signalisierung verbleibt durch den Einsatz von SCTP kompatibel zu herkömmlichen IP-Netzwerken.

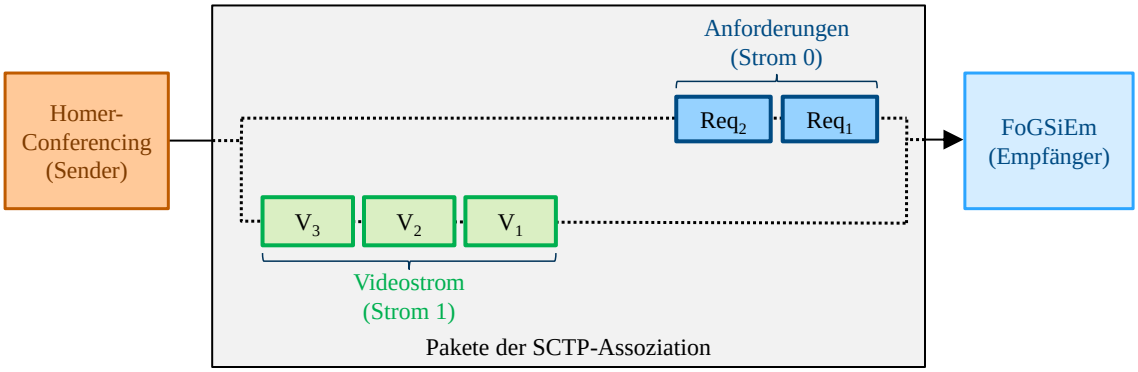

Abbildung 5.7: Videoübertragung mit Anwendungsanforderungen von Homer-Conferencing nach FoGSiEm

Abbildung 5.7 zeigt ein Beispiel einer Videoübertragung von Homer zu FoGSiEm auf Basis einer SCTP-Assoziation. Dabei wird grundsätzlich zwischen den Strömen 0 und 1 unterschieden. Ersterer dient exklusiv für die Übermittlung von Anwendungsanforderungen, während der Videostrom mit der nächst höheren Stromnummer 1 übertragen wird. Auf Seiten von FoGSiEm werden die Pakete beider Ströme ausgewertet und unterschiedlich verarbeitet. Einerseits werden die Qualitätsanforderungen der Anwendung gewonnen und für Routinganfragen innerhalb des FoG-Netzwerks verwendet. Andererseits werden die Pakete mit den Videodaten extrahiert und durch das FoG-Netzwerk zum Ziel übertragen.

5.6 Softwaremodul *Monitor*

Das Softwaremodul *Monitor* dient der Überwachung von Datenströmen und Threads der lokalen Anwendungsinstanz. Dafür stellt das Modul die notwendigen Softwaresensoren sowie die grafischen Dialoge zur Auswertung der ermittelten Werte bereit.

[17] http://lksctp.org/

[18] Im Kontext von SCTP sind sowohl verbindungslose als auch verbindungsorientierte Übertragungen möglich. Daher kann an dieser Stelle nicht explizit von Verbindungen gesprochen werden. Für SCTP wird der Begriff der *Assoziation* verwendet, um einen Datenstrom zwischen zwei Anwendungsinstanzen zu bezeichnen.

5.6.1 Messung audiovisueller Datenströme

Zur Untersuchung der Charakteristik audiovisueller Datenströme über einen längeren Zeitraum wurden gezielt Softwaresensoren in Homer eingebaut. Sie erlauben es, automatisch Statistiken für ausgehende und eingehende Datenströme zu generieren, die alle Paketdaten ab Schicht 3 des OSI-Modells beachten. Dazu zählen folgende Werte:

- **Minimale Paketgröße:** Größe des kleinsten übertragenen Pakets
- **Durchschnittliche Paketgröße:** durchschnittliche Paketgröße seit Beginn der Übertragung
- **Maximale Paketgröße:** Größe des größten übertragenen Pakets
- **Datengröße:** Gesamtgröße aller übertragenen Pakete seit Beginn der Übertragung
- **Pakete:** Gesamtanzahl von Paketen seit Beginn der Übertragung
- **Verluste:** Anzahl von verlorenen Paketen seit Beginn der Übertragung (Dieser Wert steht nur für eingehende Datenströme zur Verfügung. Er wird auf Basis der in Abschnitt 5.4.3 beschriebenen Neuimplementierung der RTP-Verarbeitung auf Empfängerseite ermittelt. Dabei werden die Sequenznummern von erfolgreich empfangenen RTP-Paketen ausgewertet und bei Lücken die verlorenen Pakete ermittelt. Der Wert ermöglicht Rückschlüsse auf die Qualität der aktuell verwendeten Route des jeweiligen Datenstroms.)
- **Momentane Datenrate:** aktuelle durchschnittliche Datenrate des Datenstroms (Der Wert wird kontinuierlich für die letzten *n* empfangenen Pakete ermittelt.)
- **Datenrate:** durchschnittliche Datenrate seit Beginn des Datenstroms

Alle Statistiken werden innerhalb der grafischen Oberfläche von Homer in einem separaten Fenster als Tabelle dargestellt. Anhang E.3 zeigt dies anhand von Beispielbildern der Anwendung. Auf Basis dieser Statistikwerte kann das Verhalten eines Videodatenstroms in Abhängigkeit von Bildveränderungen beobachtet werden. Bei kurzzeitig höheren Datenraten kann hierdurch eine eventuell verursachte Überlastsituation im Netzwerk erklärt werden. Aufgrund der eingesetzten GAPI-Schnittstelle kann dies sowohl für IP- als auch FoG-basierende Netzwerk betrachtet werden. Des Weiteren können die Statistiken für die Bestimmung von Qualitätsanforderungen in Abhängigkeit von den Einstellungen der Verarbeitungsstrecke des Softwaremoduls *Multimedia* eingesetzt werden.

5.6.2 Beobachtung der lokalen Systembelastung

Innerhalb einer Videokonferenzanwendung ist es besonders wichtig, die Laufzeiten von Abläufen gering zu halten. Dadurch werden Überlastungen des lokalen Systems und somit auch fehlerhafte Messungen von Datenströmen vermieden. Innerhalb des Softwaremoduls *Monitor* wird die Ressourcennutzung aller Threads der Anwendung überwacht. Darunter zählen sowohl der Speicherverbrauch als auch die verursachte Prozessorauslastung. Mit Hilfe der zusätzlich integrierten grafischen Anzeigen werden Lastprobleme in Echtzeit ersichtlich.

5.7 Integration für das hierarchische Routingmanagement

Die Bibliotheken von Homer wurden zusätzlich als Basis für die Integration von Videostreaming innerhalb der Software FoGSiEm verwendet. Dabei wurde der ursprüngliche Gedanke an Systemunabhängigkeit für Homer aus Abschnitt 5.1.1 aufgegriffen, sodass die resultierende Lösung in FoGSiEm ebenfalls für Linux, Windows als auch OS X zur Verfügung steht. Die Implementierung umfasst etwa 3500 Zeilen Quellcode, welche sich über 43 Dateien verteilen und eine Gesamtgröße von etwa 0,6 MB besitzen. Als Resultat kann Videostreaming unter Verwendung von beliebigen Routingimplementierungen innerhalb von FoGSiEm eingesetzt und verglichen werden. Dabei können die resultierenden Videobilder auf Empfängerseite qualitativ miteinander verglichen und subjektiv bewertet werden. Die implementierte Lösung für Videostreaming in FoGSiEm wurde bereits zur öffentlichen Vorführung [132] [133] [134] eingesetzt.

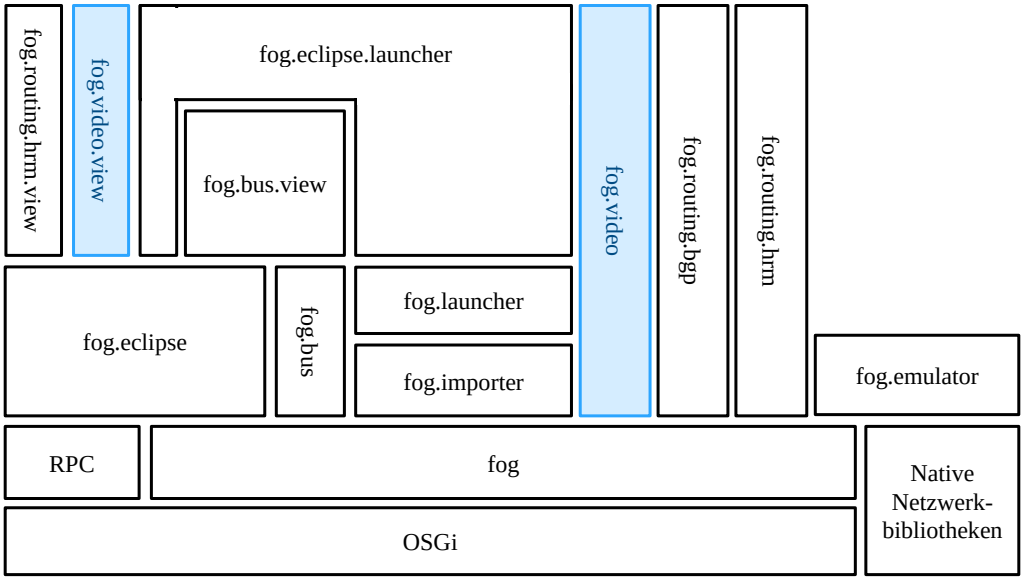

Abbildung 5.8: Erweiterung der Softwarearchitektur FoGSiEm um Videoverarbeitung und Videowiedergabe

Die Erweiterung der Software FoGSiEm erstreckt sich über die zwei Plug-Ins *fog.video.view* und *fog.video*. Die Abhängigkeiten von beiden Plug-Ins sind in Abbildung 5.8 im Kontext der FoGSiEm Softwarearchitektur dargestellt. Ein Plug-In hängt dabei funktional von allen darunter dargestellten Plug-Ins ab. Die hinzugefügten beiden Plug-Ins besitzen folgende Aufgaben:

- **fog.video.view:** Dieses Plug-In enthält ein spezielles Darstellungsfenster für FoGSiEm zur Wiedergabe eines empfangenen Videostroms in Echtzeit. Zusätzlich zeigt es Statistiken an, welche mit Hilfe des Softwaremoduls *Monitor* von Homer ermittelt werden.
- **fog.video:** Innerhalb dieses Plug-Ins sind die Funktionen zur Videoverarbeitung enthalten.

Durch die Aufteilung in zwei Plug-Ins wird die Videowiedergabe von der eigentlichen Videoverarbeitung getrennt. Für Letzteres werden die Funktionen des Softwaremoduls *Multimedia* von Homer verwendet.

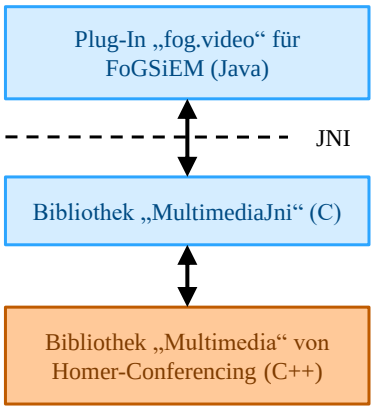

Abbildung 5.9: Nutzung der Bibliotheken von Homer-Conferencing in FoGSiEm

Abbildung 5.9 zeigt die verwendete Softwarearchitektur: In FoGSiEm werden mit Hilfe des *Java Native Interfaces* (JNI)[19] die Funktionen der Softwaremodule von Homer innerhalb der Java-Umgebung zur Verfügung gestellt. Die zusätzliche Bibliothek *MultimediaJni* wird dabei als Zwischenschicht benötigt.

[19] Durch JNI entsteht eine Brücke zwischen dem Adressraum von Java und dem von C/C++, sodass Funktionsaufrufe und Datentransfers in beide Richtungen erfolgen können.

Sie dient insbesondere zur Konvertierung von Datenformaten, sodass die folgenden Funktionen von Homer in unveränderter Form für das Plug-In *fog.video* verwendetet werden können:

- **Videopufferung:** Diese Funktion kann zum Zwischenpuffern verwendet werden, sodass *Stalling* während der Wiedergabe eines in Echtzeit übertragenen Videostroms untersucht werden kann [145].
- **Videotranskodierung:** Ein Videostrom kann neukodiert werden, sodass beispielsweise eine Konvertierung des von Codec H.264 nach H.261 möglich ist [134].
- **Videodekodierung:** Für die Wiedergabe innerhalb des Plug-Ins *fog.video.view* muss der Videostrom dekodiert werden. Als Resultat liegen aufeinanderfolgende Vollbilder im Format RGB32 vor, welche direkt wiedergegeben werden können.

Weitere Details zum Videostreaming in FoGSiEm und der grafischen Videodarstellung sind im Anhang D.4 zu finden.

5.8 Diskussion der Implementierung

Die Implementierung setzt alle in Abschnitt 5.1.1 aufgestellten Anforderungen an ein gewünschtes Multimediatestbett um. Sie kann als eigenständige Lösung für Videokonferenzen ohne zusätzliche Software eingesetzt werden. Insbesondere lag die Unterstützung verschiedener Codec- und Qualitätseinstellungen im Vordergrund der Implementierungsarbeiten. Es können explizite audiovisuelle Datenströme gesendet und empfangen werden. Des Weiteren setzt die Implementierung auf das bekannte Konzept der GAPI-Schnittstelle zur Verallgemeinerung der Schnittstellen verschiedener Netzwerkstacks. Dies kann für einen direkten Vergleichen unterschiedlicher Netzwerk- und Routingkonzepte eingesetzt werden. Für jeden Datenstrom können dabei mit Hilfe der Software die expliziten Qualitätsanforderungen über grafische Dialoge festgelegt werden. Diese werden an den Netzwerkstack signalisiert, um ein QoS-Routing zu ermöglichen. Zusätzlich generiert die Software pro Datenstrom automatisch Statistiken, sodass eine Charakterisierung mit Fokus auf notwendige Qualitätsanforderungen für die Übertragung durch das Netzwerk erleichtert wird. Sollten dennoch Paketverluste während der Übertragung auftreten, kann die Software diese mit Hilfe des integrierten Protokolls RTP erkennen. Hierüber lassen sich Aussagen über die Güte der aktuellen Route ableiten, sodass ein Vergleich zwischen verschiedenen Routingalgorithmen möglich wird. Durch die Software wird anschaulich demonstriert, dass audiovisuelle Datenströme der gleichen Quelle über verschiedene Routen übertragen werden können, ohne signifikante Auswirkungen auf die Qualität der Anwendung zu verursachen. Dies ist insbesondere im Kontext des HRM-Konzeptes aufgrund seiner QoS-orientierten Routingentscheidungen interessant.

5.9 Schlussfolgerungen

Kapitel 5 enthält einen Überblick über die Software Homer-Conferencing [11]. Sie stellt den zweiten praktischen Teil dieser Arbeit dar. Die Software ist als eigenständige Lösung für Videokonferenzen verwendbar. Als besondere Vorteile der Software sind zu nennen:

- **Explizite Generierung und Übertragung von audiovisuellen Datenströmen:**
    - Homer eignet sich für die Untersuchung von Übertragungsqualitäten insbesondere durch seine zahlreichen **Konfigurationsmöglichkeiten**. Auf Senderseite können die Charakteristiken jedes Stroms, wie beispielsweise der verwendete Codec sowie die Bildauflösung, durch grafische Dialoge eingestellt werden.
    - Die Software erlaubt des Weiteren für ausgewählte audiovisuelle Datenströme die Definition und Übertragung von **Qualitätsanforderungen**, welche durch das Routing im Netzwerk für eine akzeptable Übertragung beachtet werden müssen.

    - Dabei kann mit Hilfe der grafischen Oberfläche zwischen verschiedenen Implementierungen eines **Netzwerkstacks** und der dabei zum Einsatz kommenden **Transportprotokolle** gewählt werden.
- **Überwachung und Wiedergabe von audiovisuellen Datenströmen:**
    - Als Gegenstück zu den zuvor erläuterten Konfigurationsmöglichkeiten auf Senderseite enthält die Software verschiedene Möglichkeiten zur Auswertung von empfangenen audiovisuellen Daten. Dazu zählt neben der Generierung von **Langzeitstatistiken** auch die grafische Ausgabe von **Messwerten in Echtzeit** auf Empfängerseite. Hierdurch sind Rückschlüsse auf die Qualität der genutzten Route möglich.
    - Parallel zur Überwachung über Messsensoren existieren **Wiedergabemöglichkeiten**, sodass jeder Datenstrom über die lokalen Lautsprecher bzw. den Monitor in Echtzeit wiedergegeben werden kann. Dadurch kann ebenfalls eine akustische bzw. optische Überwachung der Übertragungsqualität durchgeführt werden.

Allgemein ist die Software als Multimediatestbett für Vergleiche zwischen unterschiedlichen Netzwerkszenarien und Protokollen zu verstehen. Teile der Software wurden zusätzlich in den Netzwerksimulator FoGSiEm integriert, sodass eine vollständige Videostrecke für FoG-basierte Netzwerke zur Verfügung steht. Die vielfältigen Möglichkeiten der Software wurden bereits zur öffentlichen Vorstellung der Konzepte von HRM [131] als auch FoG [132] [133] [134] eingesetzt. Der Quellcode der Software steht zusätzlich für zukünftige Forschungsarbeiten als Open-Source-Lösung der Öffentlichkeit zur Verfügung [130]. Weiterhin wurde ein mögliches Szenario für die zukünftige Verwendung von Homer in [146] beschrieben. Darin wird eine IT-Infrastruktur vorgestellt, in welcher Homer zur Liveübertragung des Versuchsaufbaus sowie zur interaktiven Lehrer-Schüler-Kommunikation verwendet wird.

6 Empirische Evaluierung des hierarchischen Routingmanagements

Die Bewertung von HRM muss sowohl anhand von qualitativen als auch quantitativen Kriterien erfolgen. Dazu zählen eine Einschätzung der bereitgestellten Funktionalitäten sowie die Art ihrer Realisierung. Wie in Abschnitt 3.10.9 erläutert, werden durch die Abläufe der Kontroll- und Datenebene von HRM alle in Abschnitt 3.1 ursprünglich aufgestellte Anforderungen beachtet:

- **Kernfunktionen**
  - **Adresszuweisung:** Abschnitt 3.10.3 hat für die Adressvergabe bereits gezeigt, dass stets alle Netzwerkschnittstellen auf allen Knoten des Netzwerks eine eigene global eindeutige Adresse zugewiesen bekommen. Dabei eignet sich die hierarchische Struktur von HRMIDs insbesondere für die bei der Verteilung von Routingdaten angewandten Aggregationsmechanismen, da sie eine einfache Zusammenfassung von Routingzielen in Form von Clusteradressen ermöglichen.
  - **Verteilung von Routingdaten:** Auf Basis der vergebenen Adressen bestimmt jede platzierte Managementinstanz kontinuierlich Routingdaten, welche die aktuelle Topologie und deren QoS-spezifische Eigenschaften beschreiben. Diese Daten werden zwischen den Instanzen ständig ausgetauscht, sodass auf jedem Knoten eine Routingtabelle erstellt und kontinuierlich aktualisiert wird.
  - **Routingalgorithmus:** HRM stellt ein QoS-Routing für Anwendungsdaten zur Verfügung, welches bei jeder Routingentscheidung sowohl die Qualitätsanforderungen der Anwendung als auch die aktuellen Einträge der knotenlokalen Routingtabelle beachtet. Dadurch hängt jede ermittelte Routingentscheidung immer von der aktuellen Lastsituation im Netzwerk ab.
- **Zusätzliche Eigenschaften**
  - **Autonomität:** Die Abläufe der Kontrollebene erfolgen ohne manuelle Eingaben. Abschnitt 3.10.2 zeigt, dass die Platzierung der Managementinstanzen der Kontrollebene für eine konstante Netzwerktopologie in endlicher Zeit zu einer stabilen Lösung führt, die eine korrekte Verteilung von Koordinatoren beinhaltet. Dies konnte im Rahmen der Evaluierung durch Experimente ausnahmslos bewiesen werden.
  - **Kompatibilität:** Der Einsatz von HRM in IP-basierten Netzwerken wurde im Abschnitten 3.9 detailliert beschrieben. Für FoG-basierte Netzwerke beschreibt Kapitel 4 die Integration von HRM, deren resultierende Implementierung die Grundlage für dieses Kapitel bildet.
  - **Skalierbarkeit:** Zur Unterstützung der Skalierbarkeit wird das Netzwerk in Cluster mit begrenztem Durchmesser unterteilt. Die Cluster werden wiederum mit Hilfe von hierarchisch angeordneten Koordinatoren verwaltet und dabei sowohl Adressen als auch Routingdaten im Netzwerk verteilt.
  - **Modularität:** Das Konzept von HRM unterscheidet strikt zwischen Kontroll- und Datenebene. Jegliche Prozesse der Kontrollebene laufen unabhängig voneinander ab und bilden die Basis für die Abläufe der Datenebene – die Implikationen zwischen den einzelnen Komponenten der HRM-Architektur sind in Abschnitt 3.10.4 beschrieben.

Der Vorteil der Signalisierungen von HRM zur Verteilung von Managementinstanzen, Adressen und Routingdaten liegt vor allem in der autonomen Arbeitsweise. Alternative Protokolle, welche ähnliche Daten im Netzwerk verteilen, unterscheiden sich aufgrund der verwendeten manuellen Eingaben stark in ihren verwendeten Annahmen und ihrer Funktionsweise. Somit ist es auch schwierig, geeignete Gemeinsamkeiten zu finden, welche für einen aussagekräftigen quantitativen Vergleich zwischen ihnen und HRM genügen. Des Weiteren ist die Behandlung von QoS-spezifischen Linkeigenschaften bei häufig verwendeten Routingprotokollen kein Kernthema und wird nur durch entsprechende Erweiterungen (siehe Abschnitt 2.2.6) abgedeckt, deren Einsatz eher eine Ausnahme für heutige Netzwerke darstellt

und deren Konzeptionen nicht alle Aspekte von HRM abdecken (teils fehlen auch wichtige Konzeptbeschreibungen). Folglich wurde ein direkter Vergleich von HRM mit klassischem OSPF bzw. BGP sowie seinen zugehörigen Erweiterungen vermieden. Stattdessen konzentriert sich die empirische Evaluierung dieser Arbeit vor allem auf eine möglichst allgemeingültige Bewertung der Kosten und des Nutzens, welche sich aus der Verwendung von HRM ergeben. Darüber hinaus ist es ebenfalls wichtig, mögliche Einschränkungen für den Einsatz von HRM zu identifizieren. Zusammenfassend beinhaltet die empirische Evaluierung in diesem Kapitel Untersuchungen zu folgenden Fragestellungen:

- **Welcher Signalisierungsaufwand entsteht durch die Kontrollebene?** Insbesondere die verwendete Topologie sowie der gewählte Clusterradius spielen dabei eine maßgebende Rolle. Zur Beantwortung der Frage bietet es sich an, zwischen der Startphase und dem laufenden Betrieb des Netzwerks zu unterscheiden. Dadurch werden die einmaligen Kosten zur Initialisierung von den kontinuierlichen Kosten des Routingmanagements separiert betrachtet. Als Bewertungskriterien sind dabei sinnvoll:
    - Startphase:
        - Anzahl von ausgetauschten Nachrichten zwischen den Knoten: Jede Signalisierungsart verwendet ihren eigenen Nachrichtentyp. Je mehr Übertragungen stattfinden, desto länger dauert die Initialisierungsphase des Hierarchieaufbaus oder der Adressvergabe. Durch Zählen der im Netzwerk übertragenen Nachrichten bis zum Erreichen der finalen Lösung für die Hierarchiebildung bzw. bis zur vollständigen Adressvergabe bei Variation der verwendeten Eingabeparameter wird ein Vergleich der Dauer des Initialisierungsvorgangs möglich.
    - Betriebsphase:
        - Durchschnittliches Signalisierungsaufkommen an einem Netzwerklink aufgrund der Signalisierungen der Kontrollebene: Je mehr Übertragungen nach Abschluss der Startphase weiterhin zur kontinuierlichen Aktualisierung der Hierarchie- und Routingdaten während des Netzwerkbetriebs zum Einsatz kommen, desto mehr Linkkapazitäten stehen nicht für Anwendungsdaten zur Verfügung.

  Bei den Messungen werden folgende Einflussfaktoren näher betrachtet:
    - Clusterradius: Der Wert 0 und sehr große Clusterradien stellen dabei die Extremfälle dar und sind insbesondere interessant für die Abschätzung des Signalisierungsaufwands.
    - Netzwerktopologie: Durch gezielte Auswahl von geeigneten Netzwerkstrukturen muss das Verhalten der Kontrollebene mit steigender Knotenanzahl untersucht werden. Daraus lässt sich die Skalierbarkeit der Kontrollebene für eine steigende Netzwerkgröße ableiten. Nähere Details zu den ausgewählten Topologien werden in Abschnitt 6.1 gegeben.
    - Clusterunterteilung und Zielaggregation: Durch Aggregation von topologischen Details wird bei der Verteilung von Routingdaten das verursachte Datenaufkommen reduziert. Dadurch kann eine aggregierte Route für verschiedene Zielknoten des zugehörigen Zielclusters verwendet werden. Dies hat insbesondere bei komplexeren Netzwerken Einfluss auf den resultierenden Signalisierungsaufwand während der Betriebsphase.
    - Hierarchietiefe: Eine Hierarchietiefe von 1 erscheint dabei nicht sinnvoll, stattdessen sind größere Werte interessant. Der gewählte Wert wirkt sich insbesondere auf die Betriebsphase aus und beeinflusst dabei das kontinuierliche Signalisierungsaufkommen aufgrund der unterschiedlichen Struktur der Kontrollebene.
    - Signalisierungsintervalle: Insbesondere die Nachrichten für die Koordinatorbekanntgabe und Verteilung von Routingdaten werden während des Netzwerkbetriebs kontinuierlich versendet und sind maßgebend für die verursachten Kosten der Kontrollebene.

Je häufiger Nachrichten eines Typs versandt werden, desto mehr Netzwerkressourcen stehen dadurch nicht für Anwendungsdaten zur Verfügung.

- **Welcher Speicheraufwand entsteht durch die Kontrollebene?** Die Untersuchung des Signalisierungsaufwands konzentriert sich auf die Belastung von Links, dagegen stehen bei der Betrachtung des Speicheraufwands die Knoten des Netzwerks im Vordergrund. Dabei eignen sich folgende Bewertungskriterien:
  - Anzahl von notwendigen Kommunikationsverbindungen: Die Kontrollebene verwendet feste Verbindungen zwischen den Knoten, um stetig Signalisierungen für die Verteilung von Routingdaten und zur Aufrechterhaltung der Hierarchie zwischen den Entitäten auszutauschen.
  - Größe des HRGs eines Knotens: Entsprechend Abschnitt 4.2.6 wird ein Routinggraph für die Berechnung von Signalisierungsdaten zur Verteilung von Routingdaten benötigt, welcher mit zunehmender Netzwerkgröße ebenfalls wächst.

  Bei beiden Kriterien sind sowohl die Extremfälle (Minimum und Maximum) als auch der durchschnittliche Wert interessant. Ihre Ausprägung wird wiederum durch folgende Eingabeparameter beeinflusst:
  - Clusterradius: Ähnlich dem resultierenden Signalisierungsaufkommen ist auch die verursachte Speicherbelastung abhängig vom verwendeten Clusterradius. Dabei sind der Wert 0 und auch sehr große Werte besonders interessant.
  - Netzwerktopologie: Die Struktur des Netzwerks beeinflusst sowohl die Anzahl von Kommunikationsverbindungen als auch die der Einträge im lokalen HRG eines Knotens. Dabei ist es wichtig zu untersuchen, mit welchem Maß der Speicheraufwand mit steigender Knotenanzahl zunimmt.
- **Welche Netzwerkressourcen sind auf Basis der Datenebene nutzbar?** Dies muss anhand von ausgewählten Szenarien durch einen einen Vergleich zwischen HRM- und BE-basiertem Routing mit quantitativen Ergebnissen belegt werden. Da der Fokus dieser Arbeit auf dem IntServ-Modell liegt, bietet sich als Bewertungskriterium an:
  - Anzahl von erfolgreichen Reservierungen: Durch Vergleich der erfolgreichen Verbindungsversuche wird eine Abgrenzung zwischen beiden System möglich, sodass der Nutzen von HRM gegenüber heutigem BE-Routing belegt wird.

  Als maßgebende Einflussgrößen sind dabei interessant:
  - Clusterradius: Durch sehr kleine und sehr große Werte können sich Cluster mit unterschiedlicher Größe innerhalb der Hierarchie ausbilden. Die resultierenden Clustergrößen bestimmen dabei den Detailgrad der signalisierten Routingdaten und somit auch den Inhalt der Routingtabellen. Dadurch sind Auswirkungen auf die letztlich ausgeführten Routingscheidungen und den erzielten Nutzen von HRM zu vermuten.
  - Routingschleifen: Wenn die Signalisierung von Routingschleifen generell vermieden wird, existieren in den Routingtabellen keine Einträge über die betroffenen Routen. Dies kann nachteilig für den durch HRM erzielten Nutzen sein. Eine explizite Signalisierung von ausgewählten Routingdaten über vorhandene Routingschleifen kann somit sinnvoll sein und muss näher untersucht werden.
  - Netzwerktopologie: Die Struktur des Netzwerks legt die Anzahl von redundanten Routen zu einem Ziel fest und wirkt sich somit auch auf den von HRM gegenüber BE-Routing erzielten Vorteil aus.

  Des Weiteren ist im Kontext der Betrachtungen zur Datenebene die Frage interessant, inwiefern die angewandten Aggregationsmechanismen die Menge von nutzbaren Netzwerkressourcen beeinflussen. Solche Einschränkungen bei der Routenwahl können dagegen im Rahmen von Netzwerkrichtlinien auch ausdrücklich erwünscht sein. Die Möglichkeiten zur Umsetzung von Netzwerkrichtlinien beim Einsatz von HRM müssen ebenfalls untersucht werden.

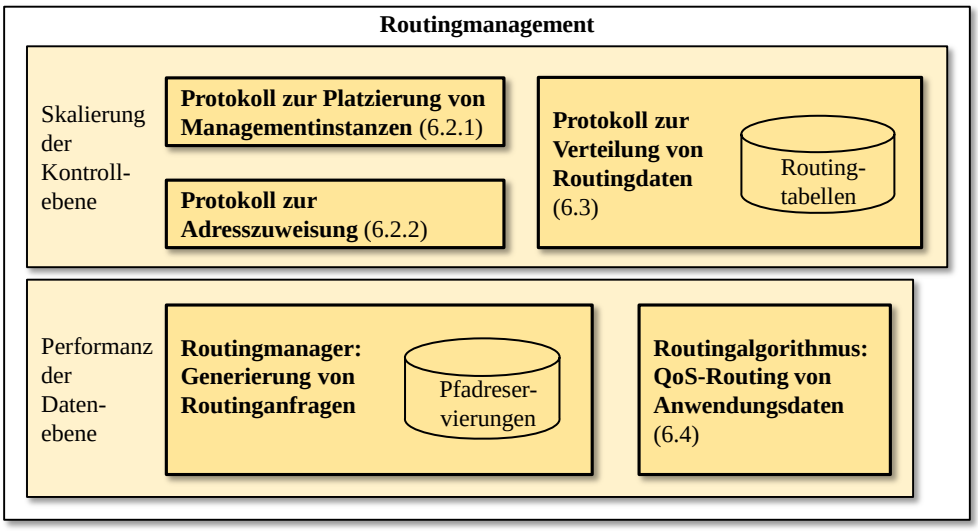

Abbildung 6.1: Untersuchung der einzelnen Komponenten der Architektur

Abbildung 6.1 zeigt die in Abschnitt 3.2 beschriebene Architektur von HRM und führt in Klammern auf, in welchem der nachfolgenden Abschnitte Messergebnisse zu den jeweiligen Komponenten enthalten sind. Ähnlich der Konzeptbeschreibung wird dabei zwischen Kontroll- und Datenebene unterschieden. Zusätzlich wird zwischen der Start- und Betriebsphase eines Netzwerks unterschieden, sodass bei den resultierenden Ergebnissen zwischen einmaligen und kontinuierlichen Kosten unterschieden wird.

Alle durchgeführten Experimente wurden auf Basis der Implementierung von Kapitel 4 durchgeführt. Dabei wurde das Verhalten von HRM sowohl in Netzwerksimulationen als auch – auf Basis der Emulatorfunktion von FoGSiEm – in realen Netzwerken untersucht. Letzteres diente zur Bewertung der drei dezentral ablaufenden Protokolle der Kontrollebene in Hinblick auf Datenkonsistenz und ausreichende Synchronisation der Abläufe. Die Durchführung der nachfolgend vorgestellten Untersuchungen startete im Januar 2014 und wurde im März 2015 abgeschlossen.

6.1 Simulationssetup

Da durch die Prozesse der Kontrollebene viele nebenläufige Abläufe durchgeführt werden, ist die Anzahl der simulierten Netzwerkknoten in jedem Szenario aufgrund der limitieren Ressourcen der Simulationshardware begrenzt gehalten. Durch dieses Vorgehen werden auch Beeinflussungen der Messwerte (beispielsweise verändertes Laufzeitverhalten durch Speicherknappheit) verhindert. Anhang F beinhaltet eine Übersicht über die bei allen Experimenten verwendete Hardware.

Alle durchgeführten Experimente basieren auf der Implementierung aus Kapitel 4 und verwenden die darin vollständig Umsetzung der Signalisierungsprotokolle der Kontrollebene bzw. des Routingalgorithmus aus Kapitel 3. Diese Methodik unterscheidet die Evaluation in dieser Arbeit von anderen Ansätzen. Sie verwenden häufig eine explizite (meist zentral durchgeführt) Statusberechnung für ausgewählte Netzwerktopologien. Dadurch eigenen sie sich eher für sehr große Topologien, sie können aber Fragen zur praktischen Anwendbarkeit der Konzeption offenlassen. Dazu zählt insbesondere die Frage, ob durch die konzipierten Signalisierungsabläufe ausreichend Daten im Netzwerk verteilt werden, sodass unter realen Bedingungen (auf Basis von ausschließlich dezentral ablaufenden Signalisierungen) das Gesamtsystem ebenfalls das erwartete Ergebnis liefert. Dagegen bietet die in dieser Arbeit angewandte Vorgehensweise eine hohe Sicherheit für die Nutzbarkeit der Konzeption bzw. Implementierung für reale Netzwerke.

6.1.1 Betrachtete Netzwerktopologie

Entsprechend Abschnitt 2.2.1 ist die Unterstützung von Qualitätsanforderungen insbesondere innerhalb eines Firmen- oder Universitätsnetzwerks interessant, sodass Intra-AS-Topologien bei der Wahl geeigneter Szenarien am sinnvollsten sind. Im Gegensatz zu den auf Inter-AS-Ebene vorherrschenden Potenzgesetzen, welche beispielsweise in [147] untersucht worden sind, besitzt jedes Intra-AS-Netzwerk eine individuelle Struktur. Dies gilt insbesondere für Firmennetzwerke, deren Strukturen insbesondere durch lokale Gegebenheiten (bspw. die Lage und Anzahl von Etagen/Häusern) und ökonomischen Überlegungen (Kosten für Equipment und Verkabelung) geprägt sind. Für die Evaluierung der Kontrollebene konnte somit keine als „typisch" zu bezeichnende Topologie identifiziert werden. Stattdessen liegt den Untersuchungen die Grundannahme zugrunde, dass jedes Netzwerk aus den gleichen Basisstrukturen besteht. Dabei sind insbesondere Netzwerke mit redundanten Links bzw. Pfaden interessant, da nur sie Alternativwege beinhalten, bei denen HRM seine Vorteile zeigen kann. Bei einer einfachen Sterntopologie ohne Alternativwege hingegen degradiert HRM zu einem klassischen Routingsystem: Datenströme werden stets entlang des gleichen Pfades zum Ziel geleitet.

Bei den nachfolgend vorgestellten Messungen wird ein zentrales Core-Netzwerk angenommen, an dessen Router zusätzliche Endgeräte angeschlossen sein können und über das Core-Netzwerk miteinander kommunizieren. Endgeräte sind dabei immer Bestandteil einer separaten Broadcast-Domäne. Insgesamt werden nachfolgend die folgenden vier typischen Basisstrukturen betrachtet:

- **Core-Netzwerk**
  - **Ringtopologie:** Jeder Knoten besitzt genau zwei Nachbarn, insgesamt stellt der resultierende Netzwerkgraph einen Kreis dar. Diese Topologie ist für innere Teile des Firmennetzwerks sinnvoll, in welchem Router durch Redundanz eine möglichst zuverlässige Anbindung zueinander besitzen sollen. Dabei stellt diese Netzwerkstruktur einen guten Kompromiss zwischen Kosten (verursacht durch Hardwarekauf und Verlegung von Glasfaserkabeln) und Nutzen dar. Sie ist besonders häufig in Firmen- und Universitätsnetzwerken zu finden.
  - **Maschentopologie:** Alle Knoten besitzen bei dieser Topologie einen direkten Link zueinander, sodass sich die Anzahl möglicher Alternativwege mit steigender Knotenanzahl kontinuierlich erhöht. Neben der Ringtopologie bietet sich diese Struktur für zentrale Abschnitt eines Firmennetzwerks an, in denen eine zuverlässige Anbindung von Teilnetzwerken gegeben sein muss. Eine Untergruppe dieser Topologie sind die losen Maschen, in denen die eher teure Maschentopologie an einigen Stellen nur in vereinfachter Form angewandt wird, sodass nicht mehr jeder Knoten zu jedem anderen einen direkten Link besitzt und dadurch Kosten gespart werden.
  - **Sterntopologie:** Bei dieser Netzwerkstruktur existiert ein zentraler Router, der über jeweils zwei[1] Links eine Anbindung an die Gateway-Router von anderen Netzwerkabschnitten hat. Bei der Untersuchung dieser Topologie kann die Betrachtung auf den zentralen Router und seine benachbarten Gateway-Router beschränkt werden, da die über die Gateways angebundenen zusätzlichen Netzwerkabschnitte ausschließlich aus Endgeräten bestehen. Diese gehören – wie nachfolgend detaillierter betrachtet – einer Broadcast-Domäne an, welche separat untersucht wird.

[1] Es können auch mehr als zwei redundante Links sein, jedoch werden in dieser Arbeit Sterntopologien immer mit zwei redundanten Links betrachtet.

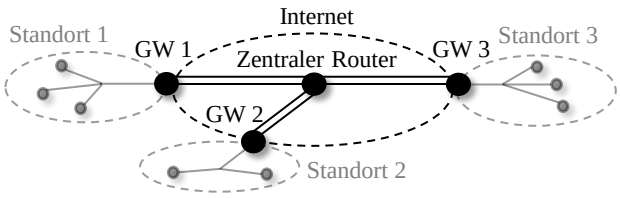

Abbildung 6.2: Sterntopologie mit einem zentralen Router und 3 angeschlossenen Standorten

Abbildung 6.2 zeigt ein solches Beispielszenario, in welchem 3 Firmenstandorte über einen zentralen Router im Internet miteinander verbunden sind. Typischerweise kommt in realen Anwendungsfällen dabei eine VPN-Software zum Einsatz, wodurch aus Sicht des Firmennetzwerks der zentrale Router zum direkten Nachbarn aller Gateway-Router der Standorte wird[2].

- **Broadcast-Domäne mit Endgeräten:** In diesem Szenario sind mehrere Knoten Mitglieder der gleichen Broadcast-Domäne und können mit Hilfe von Switches miteinander Daten austauschen. Typischerweise wird dies in eigenständigen Firmenabteilungen oder -standorten verwendet, deren Computer an einen zentralen Router angebunden sind und über ihn Verbindungen in andere Netzwerke herstellen. Aus Sicht des Routings existieren direkte Links zwischen allen Knoten einer Broadcast-Domäne. In Abbildung 6.2 wird ein solches Netzwerk beispielsweise in jedem der drei abgebildeten Standorte verwendet.

Bei der Wahl der Größe der untersuchten Szenarien spielten auch Überlegungen zur Simulationszeit eine wichtige Rolle. Da für die Messungen eine vollständige Implementierung aller konzipierten Protokolle verwendet wurde, war es wichtig, die Anzahl von untersuchten Knoten der Menge von auftretenden nebenläufigen Signalisierungen anzupassen. Ein Überblick über die verwendete Hardware ist in Anhang F zu finden. Die Knotenanzahl wurde dennoch weit genug variiert, um das Verhalten allgemein für eine steigende Topologiegröße einschätzen zu können.

6.1.2 Fixierung der Koordinatorplatzierung

Um die Skalierbarkeit der Signalisierungen der Kontrollebene für verschiedene Topologien sowohl während der Start- als auch Betriebsphase exakt bewerten zu können, darf die Anzahl der notwendigen Nachrichten nur vom chronologischen Verlauf von auftretenden Ereignissen abhängig sein. Wie in Abschnitt 3.3.2.2 beschrieben, ist bei einem Wahlvorgang auf Hierarchielevel 0 das Ergebnis jedoch sowohl von den Prioritäten der Knoten als auch ihren zugehörigen Knoten-IDs abhängig. Letztere sind lokal zu verwaltende Nummern, welche beispielsweise durch zufällig erzeugte UUIDs implementiert werden können, sodass die resultierende Verteilung der Knoten-IDs die Platzierung der L0-Koordinatoren und dadurch auch die Platzierung höherer Managementinstanzen beeinflussen kann. Dadurch ergeben sich Unterschiede sowohl für die Start- als auch Betriebsphase. Um dies zu vermeiden, wird innerhalb der FoGSiEm-spezifischen Implementierung von HRM ein global bekannter Zähler verwendet, dessen Wert bei Erzeugung eines neuen Knotens innerhalb der Simulation automatisch um 1 erhöht wird. Da bei FoGSiEm für die verwendeten Szenarien die Knoten stets nach einer festen Reihenfolge erzeugt werden, erfolgt somit die Verteilung der Knoten-IDs und dadurch auch der L0-Koordinatoren für ein Szenario stets nach einem konstanten Schema[3].

[2] Alternativ können die Standorte auch direkt miteinander verbunden werden. In dem Fall entspricht die resultierende Struktur je nach Konfiguration beispielsweise der Ring- oder Maschentopologie.

[3] Zur Kontrolle der Platzierung aller Koordinatoren für mehrere nacheinander ausgeführte Versuche bietet die Implementierung die Möglichkeit, eine Statistik der Platzierung auszugeben. Dadurch war es während den Versuchen möglich, die konstante Strukturierung der Kontrollebene zu prüfen. Dabei wurden keine Ausnahmen festgestellt.

6.1.3 Generierung von Datenströmen und zugehörigen Routinganfragen

Für die Betrachtungen der Performanz des QoS-Routings wurde die in Abschnitt 4.5 beschriebene Anwendung *HRMTestApp* verwendet, um Instanzen der Anwendung *QoSTestApp* im Netzwerk zu verteilen und zwischen diesen Instanzen zufällig Verbindungen aufzubauen. Die zum Aufbau notwendigen Routinganfragen werden dabei automatisch für jeden Knoten (*Hop-by-Hop*-Routing) durch die FoG-spezifische Paketweiterleitung generiert und an HRM weitergeleitet. Die Abläufe des Routingdienstes erzeugen automatisch die notwendigen Reservierungen für ausgehende Links eines Knotens und veranlassen die Aktualisierung der Routingdaten im Netzwerk. Somit kann mit Hilfe der simulierten Datenübertragungen die Performanz der Datenebene für zufällig erzeugte Lastsituationen im Netzwerk beobachtet werden.

6.2 Signalisierungsaufwand der Kontrollebene in der Startphase

Zur Bewertung der Startphase stehen die zwei Protokolle zur Platzierung der Managementinstanzen und zur Adresszuweisung im Vordergrund. Die dadurch im Netzwerk verteilten Daten bilden die Basis für die Signalisierungen von Routingdaten, welche erst während der Betriebsphase des Netzwerks eine zentrale Rolle spielen. Beide Protokolle der Startphase wurden bei den im Folgenden vorgestellten Experimenten nacheinander ausgeführt, sodass Zwischenlösungen der Strukturierung der Kontrollebene nicht die Signalisierungen der Adresszuweisung beeinflussen und somit beide Abläufe getrennt betrachtet werden können. Die Implementierung wurde zur Analyse der Startphase um folgende Funktionen erweitert:

- **Erkennung der finalen Managementinfrastruktur:** Da die Struktur der Kontrollebene ausschließlich bei Aktualisierungen von Prioritäten angepasst wird, ist die global eindeutige Simulationszeit der letzten Veränderung ein geeignetes Kriterium zur Erkennung der finalen Lösung. Insofern dabei eine festgelegte Zeit überschritten ist, wird die Struktur der Kontrollebene als „stabil“ angenommen und die Adressverteilung automatisch gestartet[4]. Ohne diesen Mechanismus könnten temporäre Adresszuweisungen das Ergebnis verfälschen.
- **Globale Statistik:** Über eine globale Statistikfunktion wird die Anzahl der Nachrichten in Abhängigkeit von ihrem Typ erfasst und bei Versand neuer Pakete automatisch aktualisiert. Dabei wird jede Nachricht bei ihrer Erzeugung einmalig gezählt. Des Weiteren werden die erzeugten und tatsächlich verwendeten Entitäten gezählt, um Ursachen für gemessene Signalisierungsaufkommen näher untersuchen zu können.
- **Neustart der Simulation:** Für eine automatisierte Messung mit mehreren Durchläufen ist eine korrekte Erkennung des Endes eines Versuches notwendig. Die Implementierung verwendet dafür die zuvor beschriebene Erkennung einer stabilen Managementstruktur. Als zusätzliche Bedingung werden die lokalen Warteschlangen aller Knoten geprüft, sodass ein Neustart erst bei Abarbeitung aller wartenden Pakete erfolgt.

Durch temporär auftretende vollständige Auslastung der Simulationshardware können Ausnahmen während den Messungen auftreten[5], welche sich durch leicht variierende Werte äußern. Um ihren Einfluss auf das Gesamtergebnis abzuschwächen, wurden für die Bewertung der Startphase für jede gewählte Topologie jeweils 200 Durchläufe durchgeführt und daraus die jeweiligen Mittelwerte bestimmt. Des Weiteren wurde das Ergebnis jedes Versuchsdurchlaufs automatisch durch zusätzlich implementierte

[4] Der genaue Schwellwert zum Start der Adressvergabe ist der Implementierung zu entnehmen. Zusätzlich wurde bei den in dieser Arbeit vorgestellten Experimenten geprüft, ob sich während der Adresszuweisung und der nachfolgenden Betriebsphase trotz konstanter Konnektivität im Netzwerk weitere Prioritätsveränderungen im Netzwerk ergeben, sodass dieser Ausnahmefall zuverlässig bei der Betrachtung der Messergebnisse ausgeschlossen ist.

[5] Zur Simulation von nebenläufigen Signalisierungen lief während der Versuche jede Instanz der *HRM-Controller*-Anwendung (siehe Abschnitt 4.1.2.1) in einem eigenen Thread, sodass durch das Scheduling innerhalb der Java-Umgebung Unterschiede in der Abarbeitung von Ereignissen auftreten können.

Prüffunktionen[6] verifiziert, sodass Inkonsistenzen in der Struktur der Managementinfrastruktur sowie verloren gegangene sowie endlos wartende Nachrichten generell ausgeschlossen werden können.

6.2.1 Initialisierung der Managementhierarchie

Die Bewertung der Startphase erfolgt durch Zählung folgender Signalisierungen[7]:

- **Clusterbildung**:
  - *RequestClusterMembership* (Abschnitt 3.3.4.3)
  - *RequestClusterMembershipAck, InformClusterLeft, InformClusterMembershipCanceled* (In Abschnitt 4.2.2.1 wurden diese drei Nachrichtentypen als implementierungsspezifische Verbesserung zur expliziten Signalisierung des Kommunikationsstatus eingeführt.)
- **Koordinatorenwahl**[8]:
  - *ElectionPriorityUpdate, ElectionWinner, ElectionResign* (Abschnitt 3.3.2.2)
  - *ElectionLeave* (Abschnitt 3.3.4.6)
  - *ElectionReturn* (Abschnitt 3.3.4.7)

Der Inhalt der einzelnen Nachrichten ist in Anhang B.1 dokumentiert. Ihre Anzahl wird nachfolgend für den Fall des Neustarts des kompletten Netzwerks betrachtet, sodass dadurch immer der Fall des maximalen Signalisierungsaufkommens betrachtet wird. Eine Einschätzung der notwendigen Zeit bis zur Ermittlung der finalen Struktur der Kontrollebene wird dabei nicht gegeben, da dies durch eine rein ereignisbasierte Simulation, wie sie in FoGSiEm verwendet wird, nicht zuverlässig bestimmt werden kann. Dies kann eine reale Implementierung auf physikalischen Knoten leisten, deren Umsetzung über den Rahmen dieser Arbeit hinausgeht.

Betrachtungen zum Einfügen oder Entfernen einzelner Knoten werden im Folgenden vermieden. Die dabei verursachten Signalisierungen hängen sowohl von der bisherigen Topologie, der daraus resultierenden Struktur der Kontrollebene und auch von der Position des Einfügens des Knotens ab. Eine Verallgemeinerung der verursachten Signalisierungen ist daher als äußerst schwierig einzustufen. Jedoch ist bei diesen Operationen davon auszugehen, dass das Signalisierungsaufkommen im Vergleich zum Komplettstart des Netzwerks signifikant kleiner ist und die Kontrollebene geringere Veränderungen durchläuft.

Bei den nachfolgend beschriebenen Experimenten wurden folgende Zeiten und Intervalle für *AnnounceCoordinator*-Signalisierungen verwendet:

- **Sendeintervall bei instabiler Hierarchie:** Sollte eine Koordinatorinstanz eine sehr kurze Lebenszeit besitzen, wird die Hierarchie an dieser Stelle als „instabil“ angenommen und Veränderungen in der Konnektivität für „wahrscheinlich“ angenommen. Somit wurde für den Fall bei den nachfolgenden Messungen ein sehr kleines Aktualisierungsintervall von 2 Sekunden verwendet. Erst wenn der Koordinator als stabil angenommen wird, wird ein größeres Intervall verwendet.
- **Zeit bis ein Koordinator als stabil angenommen wird:** Entsprechend Abschnitt 3.3.6.2 wird ab Erreichen dieser Lebenszeit eine Instanz als „bestätigt“ angenommen, sodass das Sendeintervall erhöht und die Rate der versendeten Nachrichten dadurch reduziert wird. Für diesen Parameter wird ein Wert von 30 Sekunden verwendet.

[6] Im Quellcode ist dies als Funktion *validateAllResults* innerhalb der Klasse *HRMController* zu finden.
[7] Die Nachrichten zur Nachbarschaftserkennung und Koordinatorenbekanntgabe sind periodisch wiederkehrende Signalisierungen und werden später bei der Bewertung der Betriebsphase analysiert.
[8] Der in Abschnitt 4.2.2.1 eingeführte *Alive*-Nachrichtentyp ist nicht beachtet, da er während keiner Messungen aufgetreten ist.

- **Sendeintervall bei stabiler Hierarchie:** Sobald ein Koordinator als „bestätigt“ gilt, wird das Sendeintervall für *AnnounceCoordinator*-Nachrichten erhöht und das Signalisierungsaufkommen dadurch reduziert. Für diesen Parameter wird ein Wert von 60 Sekunden verwendet.

6.2.1.1 Einfluss des Clusterradius am Beispiel der Ringtopologie

Die Evaluierungen beginnen mit einer Untersuchung des Einflusses des Clusterradius auf das Skalierungsverhalten der Kontrollebene, wobei die Hierarchietiefe konstant auf einem Wert von 3 gehalten wird. Als Basis dient dabei ein Szenario, welches ähnlich dem Referenzszenario aus Abschnitt 3.3 aufgebaut ist und aufgrund seiner kostengünstigen Umsetzung ein häufiger Vertreter in Core-Netzwerken ist. Um aussagekräftigere Messergebnisse zu erhalten, wurde die Anzahl von Knoten jedoch auf 12 erhöht, welche durch 12 Links wiederum zu einem Ring miteinander verbunden sind. In Abhängigkeit vom Clusterradius *r* ergeben sich dabei verschiedenartige Strukturen der Kontrollebene, wobei der interessante Wertebereich zwischen 0 bis 12 liegt. Für den Wert 0 ist zu erwarten, dass besonders viele Cluster auf Hierarchielevel 1 mit sehr geringer Mitgliederzahl existieren. Im Gegensatz dazu ist bei einem Wert von 12 aufgrund der hohen Reichweite von *AnnounceCoordinator*-Nachrichten zu erwarten, dass ein allumfassender Cluster auf Hierarchielevel 1 ausgebildet wird.

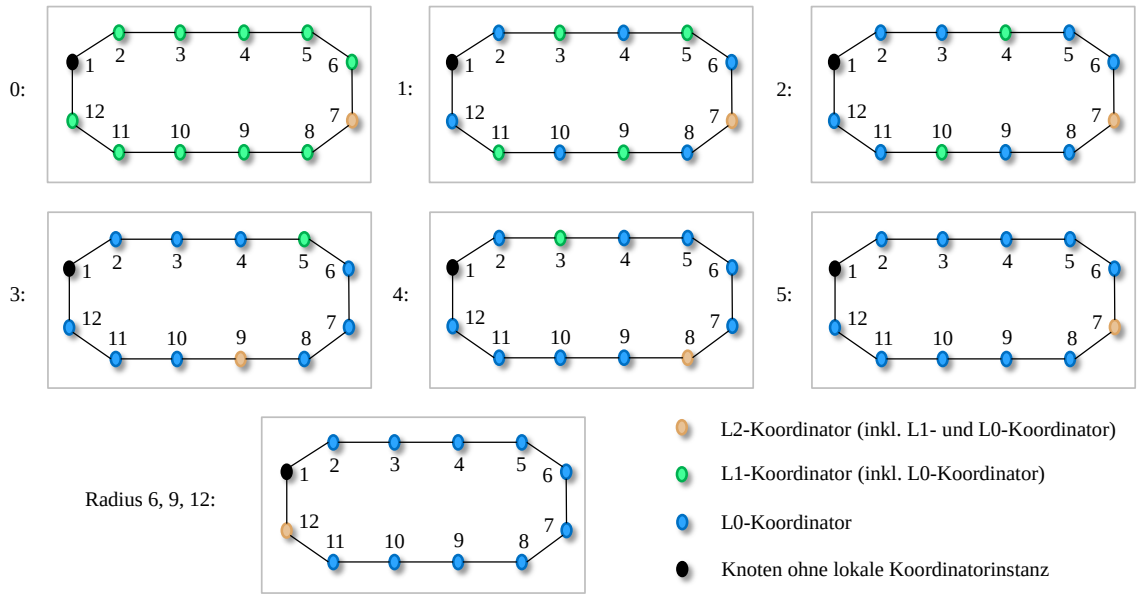

Abbildung 6.3: Platzierung der Koordinatoren im Netzwerk in Abhängigkeit vom gewählten Clusterradius

Abbildung 6.3 zeigt die resultierenden Platzierungen der Managementinstanzen, zu denen aufgrund der festgelegten Hierarchietiefe ausschließlich Koordinatoren auf den Hierarchielevels 0 bis 2 zählen. Jeder abgebildete Koordinator steht dabei ebenfalls stellvertretend für alle ihm lokal untergeordneten Instanzen der niederen Levels. Wie in den Darstellungen zu erkennen ist, werden die L0-Koordinatoren unabhängig des Clusterradius platziert, während die Lokalisierung höherer Koordinatoren variiert. Entsprechend der Erläuterungen von Abschnitt 6.1.2 ist ihre Platzierung aufgrund der vereinfachten Bestimmung der Knoten-ID von der Nummerierung der Netzwerkknoten und den in der Nachbarschaft lokalisierten Koordinatoren abhängig. Wie in der Abbildung zu sehen, sind insbesondere die resultierenden Positionen der L1-Koordinatoren unterschiedlich, wobei sie stets eine Hop-Distanz von *(r + 1)* bis *(2 * r + 1)* zueinander einhalten. Die Korrektheit entsprechend Abschnitt 3.10.2.2 ist dadurch gegeben. Der zentrale TOP-Koordinator auf Hierarchielevel 2 befindet sich aufgrund der Prioritätsverteilung zwischen den Knoten stets in zentraler Lage zu den untergeordneten L1-Koordinatoren. Allgemein gibt es zwei Extremfälle der Strukturierung: zum einen ist dies ein Radius von 0 und zum anderen alle Radien größer 5. In beiden degradiert die dreistufige Hierarchie zu einer zweistufigen. Die Auswirkungen werden nachfolgend durch Messungen verdeutlicht.

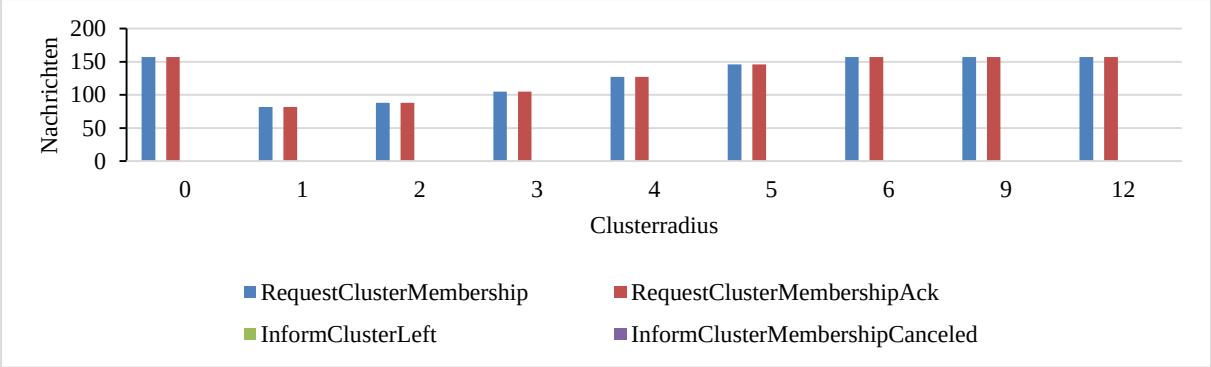

Abbildung 6.4: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Clusterbildung bei Variation des Clusterradius (Ring mit 12 Knoten)

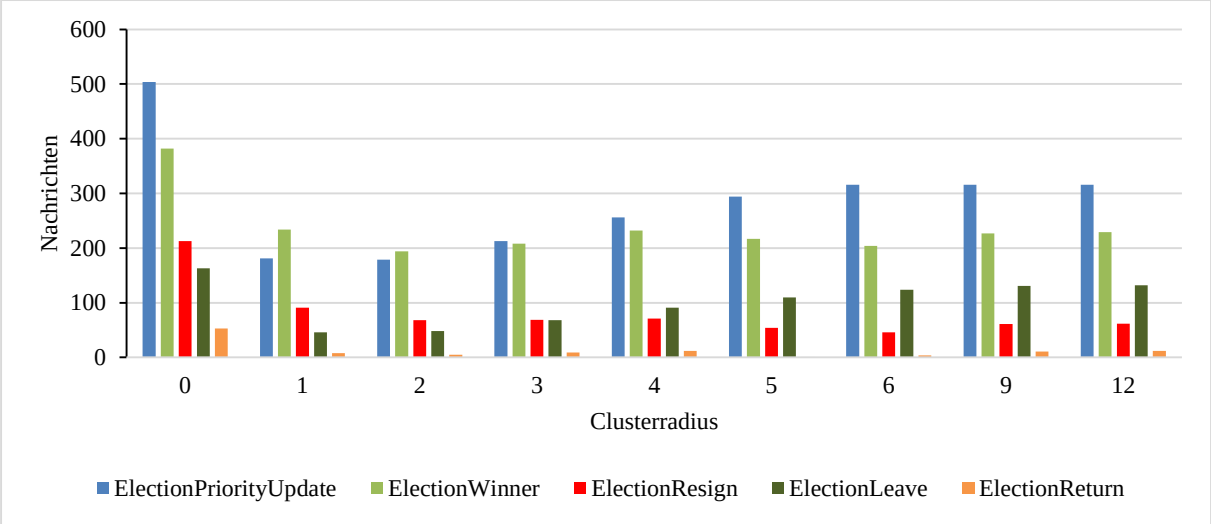

Abbildung 6.5: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Koordinatorenwahl bei Variation des Clusterradius (Ring mit 12 Knoten)

Unabhängig vom eingesetzten Clusterradius existieren 12 Links im Netzwerk, sodass dafür immer 24 L0-Clustermanager instanziiert werden, die jeweils eine *RequestClusterMembership*-Signalisierung auslösen. Durch die unterschiedliche Platzierung der höheren Koordinatoren ergeben sich die in Abbildung 6.4 dargestellten Unterschiede von notwendigen *RequestClusterMembership*- und *RequestClusterMembershipAck*-Nachrichten. Dabei sind die charakteristischen Werte 0 (große Nachrichtenanzahl), 1 (minimale Nachrichtenanzahl) und 6 (für Werte größer 5 verhält sich die Clustererstellung gleich) für den Radius zu erkennen. Diese treten ebenfalls bei den auftretenden Signalisierungen zur Koordinatorwahl auf. Wie in Abbildung 6.5 ersichtlich, sind dabei die *PriorityUpdate*- und *Winner*-Nachrichten die maßgebenden Signalisierungen.

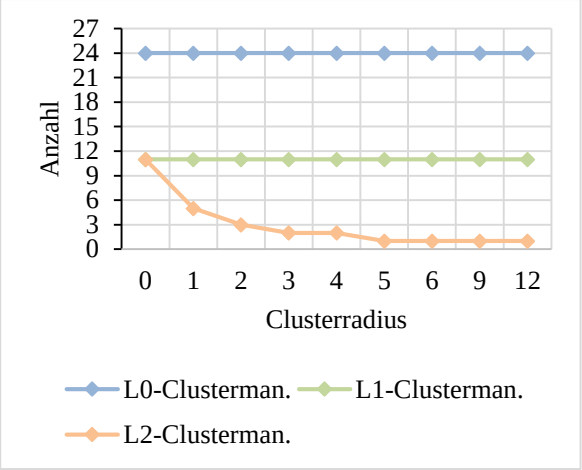

Abbildung 6.6: Clustermanager für unterschiedliche Clusterradien (Ring mit 12 Knoten)

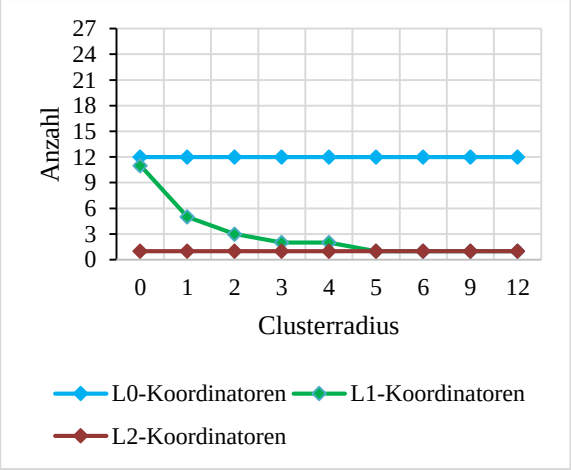

Abbildung 6.7: Koordinatoren für unterschiedliche Clusterradien (Ring mit 12 Knoten)

Bezieht man den Vergleich aus Abbildung 6.6 und Abbildung 6.7 in die Betrachtung mit ein, werden die Ursachen für die unterschiedliche Anzahl von Signalisierungen deutlich:

- Radius 0: Die Struktur beinhaltet 11 L1-Clustermanager, welche aufgrund des limitierten Ausbreitungsradius von *AnnounceCoordinator*-Nachrichten keine entfernten L0-Koordinatoren kennen. Dadurch kennt jeder nur einen untergeordneten L0-Koordinator. Die Ausnahme bildet der Clustermanager auf Knoten 12, der beide lokalen L0-Koordinatoren in seinen Cluster aufnimmt. Auf Hierarchielevel 2 werden folglich 11 L2-Clustermanager gebildet, welche jeweils zu allen untergeordneten L1-Koordinatoren eine *RequestClusterMembership*-Signalisierung durchführen. Somit ergibt sich die Gesamtzahl der Signalisierungen aus der Summe von 24, 12 und 121 zu 157.
- Radius 1: Es existieren 11 L1-Clustermanager, welche im Durchschnitt jeweils 3 untergeordnete Koordinatoren kennen und zu ihnen eine *RequestClusterMembership*-Nachricht senden. Des Weiteren besitzt jeder Knoten mit L1-Koordinator ebenfalls eine Instanz eines übergeordneten Clustermanagers, sodass sich 5 L2-Clustermanager mit jeweils 5 untergeordneten Koordinatoren ergeben. Die Anzahl der Signalisierungen ergibt sich somit aus der Summe aus 24, 33 und 25 zu 82.
- Radius 6: Für einen Wert von 6, oder größer, kennt jeder der 11 L1-Clustermanager alle 12 im Netzwerk befindlichen L0-Koordinatoren und bindet sie per *RequestClusterMembership*-Signalisierungen in seinen Cluster ein. Da Knoten 12 die höchste Knoten-ID besitzt, geht sein L1-Clustermanager als Sieger der Wahlvorgänge hervor und bildet eine Koordinatorinstanz auf Hierarchielevel 1 aus. Infolgedessen wird auf dem gleichen Knoten ein L2-Clustermanager instanziiert, der ausschließlich den lokalen L1-Koordinator in seinen Cluster aufnimmt. Folglich ergibt sich die Anzahl an Signalisierungen aus der Summe von 24, 132 (11 L1-Clustermanager mit jeweils 12 untergeordneten L0-Koordinatoren) und 1 (die Signalisierung des L2-Clustermanagers) zu 157.

Im Kontext der Clusterbildung für die erläuterte Ringtopologie führt eine Veränderung des Radius somit zu einer Steigerung des Signalisierungsaufwands um bis zu 91% (bei einem Clusterradius von 0 oder für Werte größer 5) in Bezug auf die optimale Lösung (sie stellt sich bei einem Clusterradius von 1 ein).

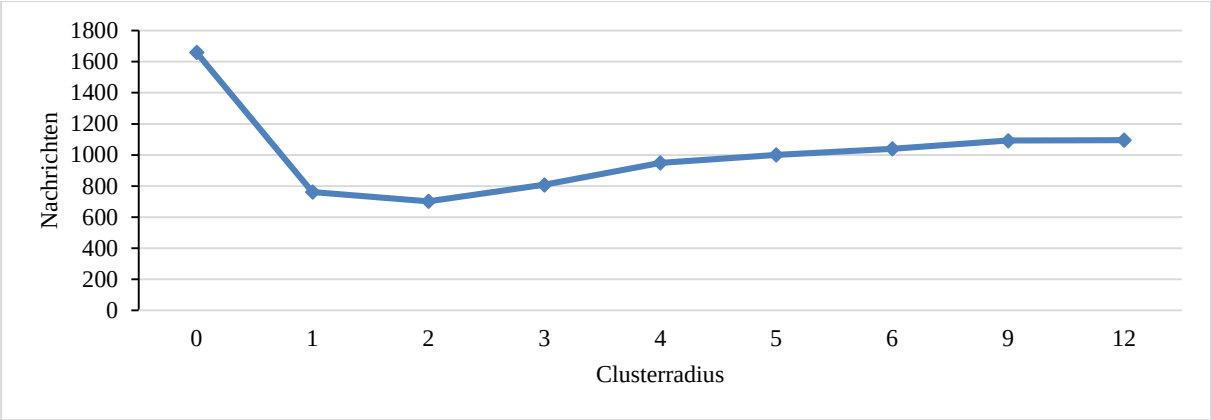

Abbildung 6.8: Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl (Ring mit 12 Knoten)

In den bisherigen Betrachtungen erscheint ein Clusterradius von 1 optimal für das Szenario. Abbildung 6.8 stellt das resultierende Signalisierungsaufkommen für die Clusterbildung und Koordinatorenwahl unter Verwendung verschiedener Clusterradien dar. Dabei scheint der Wert 2 zu dem besten Ergebnis zu führen, wobei ein Radius von 1 dagegen eine Steigerung des Aufwands um 13 % verursacht. Ein Wert von 0 führt sogar zu einer Steigerung des Nachrichtenaufkommens um rund 156 % und erscheint als besonders ungünstige Konfiguration. Alle anderen Werte (3-12) führen gegenüber der besten Lösung zu einer Steigerung von bis zu 49% und erscheinen aufgrund der sich daraus ergebenden absoluten Nachrichtenanzahl für die Startphase dennoch akzeptabel.

Allgemein wurde durch die bisherige Analyse gezeigt, dass der Wert 0 zu einer besonders schlechten Signalisierungsperformanz führt und einen ungünstigen Kandidaten für die Wahl des Clusterradius darstellt. Verursacht wird dies durch die resultierende Struktur der Hierarchie, welche einen Cluster mit einer sehr hohen Anzahl an Mitgliedern verwendet und dadurch die Signalisierungen in einer sehr flachen Hierarchie gruppiert werden. Das ist vergleichbar mit OSPF, bei dem die Router über einen zentralen *Designated Router* ihre Routingdaten austauschen und nicht direkt miteinander kommunizieren.

6.2.1.2 Einfluss der Topologie und der Anzahl von Knoten

Nachfolgend wird die Initialisierungsphase der Koordinatorplatzierung für unterschiedliche Topologien bei einer steigenden Anzahl von Knoten näher untersucht. Dabei ist der Clusterradius bei allen Messungen mit einem eher hohen Wert von 8 konstant. Er wurde so gewählt, um auch bei einer hohen Knotenanzahl eine gute Skalierung der Signalisierungen zu erhalten. Dabei war insbesondere die Ringtopologie maßgebend, da sie bis zu einer Größe von 120 Knoten untersucht wurde.

Ringtopologie

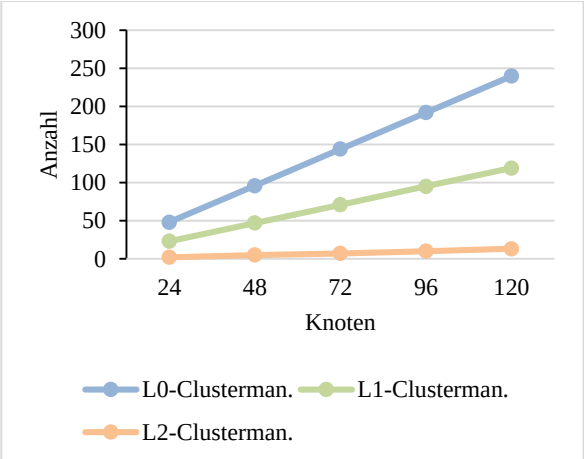

Abbildung 6.9: Erstellte Clustermanager (Ring)

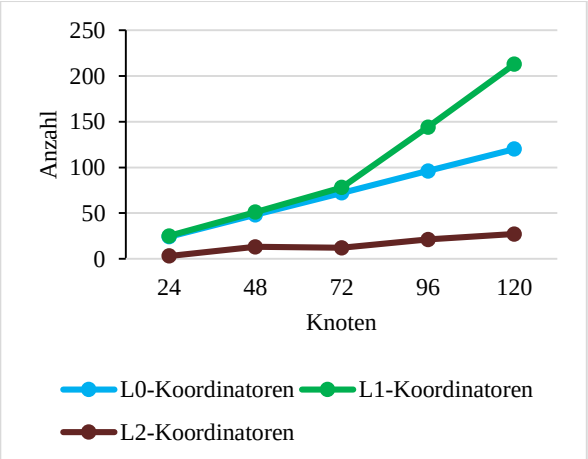

Abbildung 6.10: Erstellte Koordinatoren (Ring)

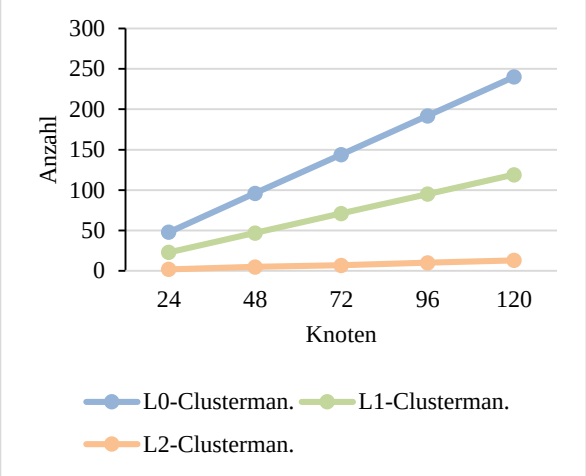

Abbildung 6.11: Verbleibende Clustermanager (Ring)

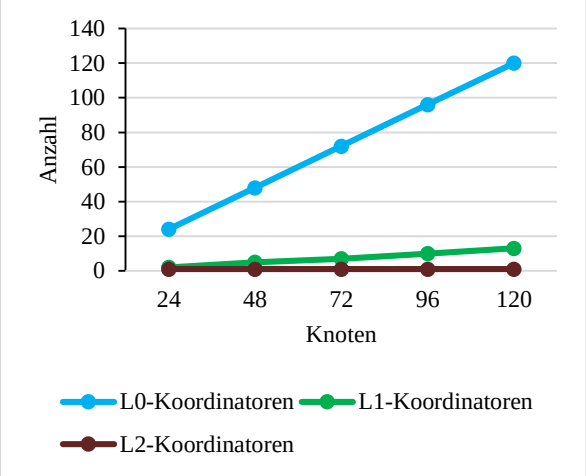

Abbildung 6.12: Verbleibende Koordinatoren (Ring)

Bei Analyse der Ringtopologie fällt in Abbildung 6.9 die lineare Zunahme von erstellten Clustermanagern auf allen Hierarchielevels auf. Dabei entspricht die Anzahl der L0-Clustermanager immer dem Doppelten der Knotenanzahl. Abbildung 6.10 zeigt die dazu gehörige Anzahl von erstellten L0-Koordinatoren – sie entspricht wiederum der jeweiligen Knotenanzahl, da für jeden Link eine eigene Managementinstanz für Hierarchielevel 0 erstellt wird. Des Weiteren verdeutlicht die Grafik, dass auf den Hierarchielevels 1 und 2 einige Koordinatorinstanzen nur temporär bestanden und dann wieder entfernt worden sind. Dies wird insbesondere anhand der Zahlen für den TOP-Koordinator deutlich: Trotz der unterschiedlichen temporären Lösungen muss dieser in jedem Netzwerk bei der finalen Lösung einzigartig sein. Zum besseren Vergleich zeigen Abbildung 6.11 und Abbildung 6.12 zusätzlich die im Netzwerk verbleibenden Entitäten der Kontrollebene und verdeutlichen, dass mit zunehmender Netzwerkgröße die Anzahl von L1- und L2-Koordinatoren eher sehr gering ansteigt, während die Menge von L0-Koordinatoren für diese Topologie der Knotenanzahl entspricht.

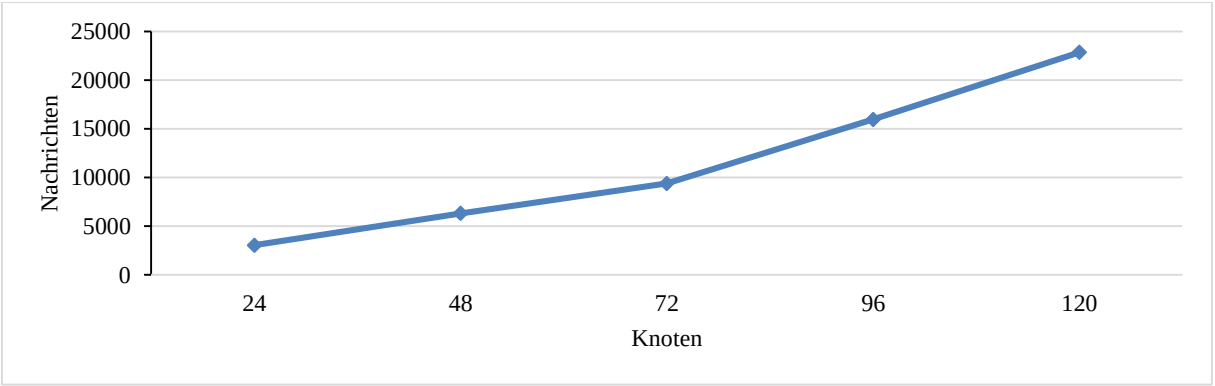

Abbildung 6.13: Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl (Ring)

Abbildung 6.14: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Clusterbildung (Ring)

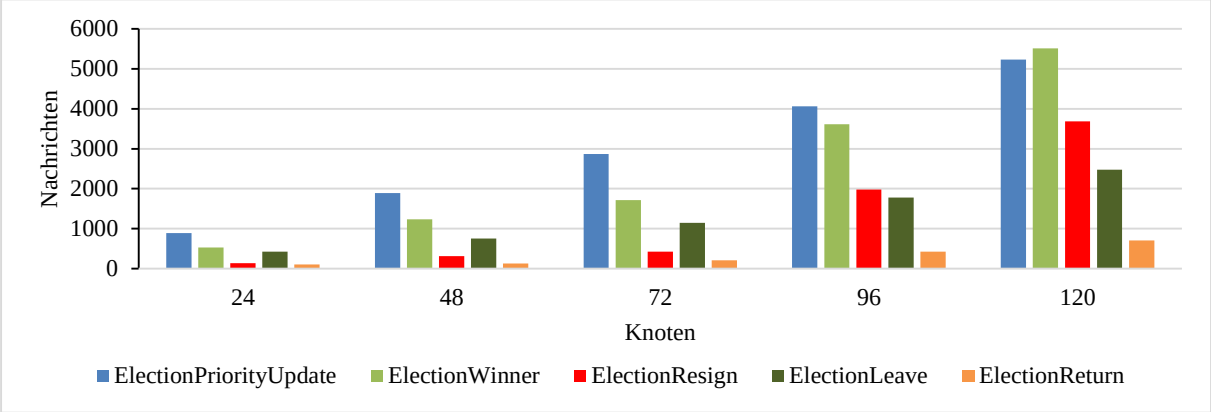

Abbildung 6.15: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Koordinatorenwahl (Ring)

Abbildung 6.13 bestätigt die lineare Zunahme von Signalisierungsaktivität und zeigt zugleich den unterschiedlichen Verlauf rund um den Wert 72, der sich aufgrund der resultierenden unterschiedlichen Strukturierung der Kontrollebene ergibt. Abbildung 6.14 und Abbildung 6.15 zeigen zudem[9], dass das Signalisierungsaufkommen vor allem durch die *PriorityUpdate*- und *Winner*-Nachrichten beeinflusst wird und die Anzahl der Zwischenlösung mit zunehmender Netzwerkgröße steigt (ersichtlich anhand der *Resign*-Nachrichten und der Aktivität des DCE-Algorithmus). Dieses Verhalten wird jedoch durch den Charakter der betrachten Topologie verursacht. Sie führt dazu, dass alle Knoten immer die gleiche L0-Priorität besitzen und erst nach Durchlauf von einigen Zwischenlösungen anhand der Knoten-IDs eine finale Struktur für die Kontrollebene gefunden wird.

[9] Die Messung von zusätzlichen Clusterradien wurde an dieser Stelle vermieden, da sich dabei wiederum das Verhalten aus Abschnitt 6.2.1.1 zeigt. Ein Clusterradius von 0 und Werte größer dem jeweiligen Netzwerkdurchmesser (entspricht für die Ringtopologie der Hälfte der Knotenanzahl) führen zu erhöhtem Signalisierungsaufkommen, während die geringste Anzahl von Nachrichten für ein Clusterradius zwischen 0 und dem Wert des Durchmessers auftritt.

Maschentopologie

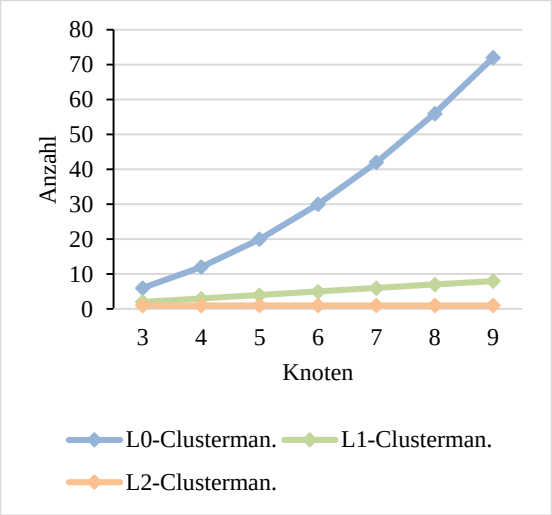

Abbildung 6.16: Clustermanager (Masche)

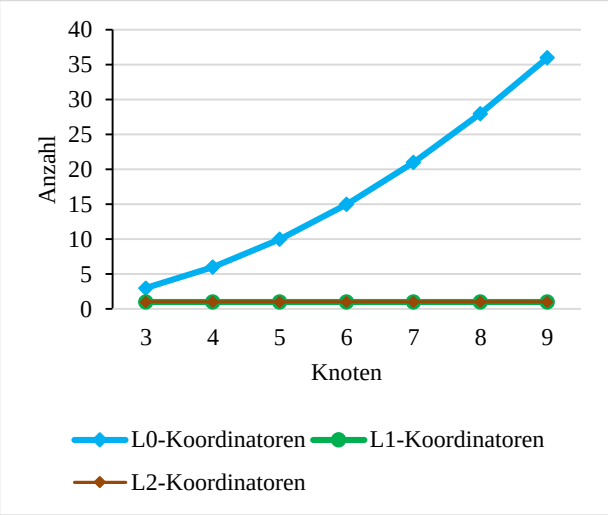

Abbildung 6.17: Koordinatoren (Masche)

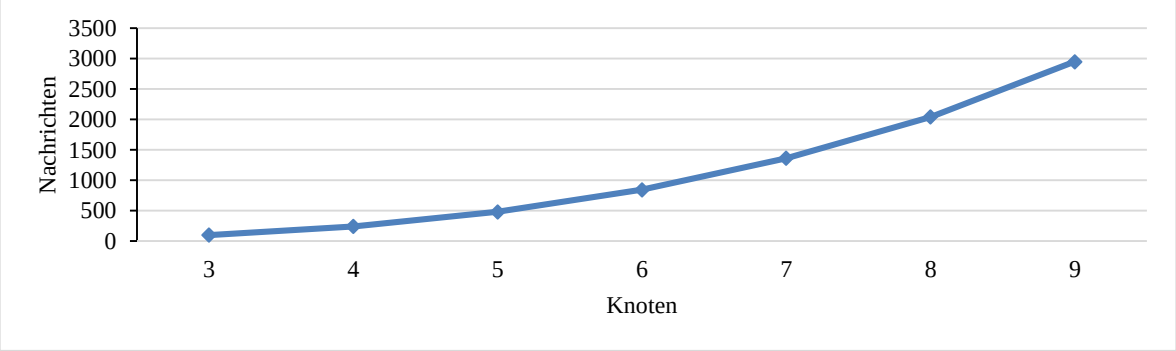

Abbildung 6.18: Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl (Masche)

Abbildung 6.19: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Clusterbildung (Masche)

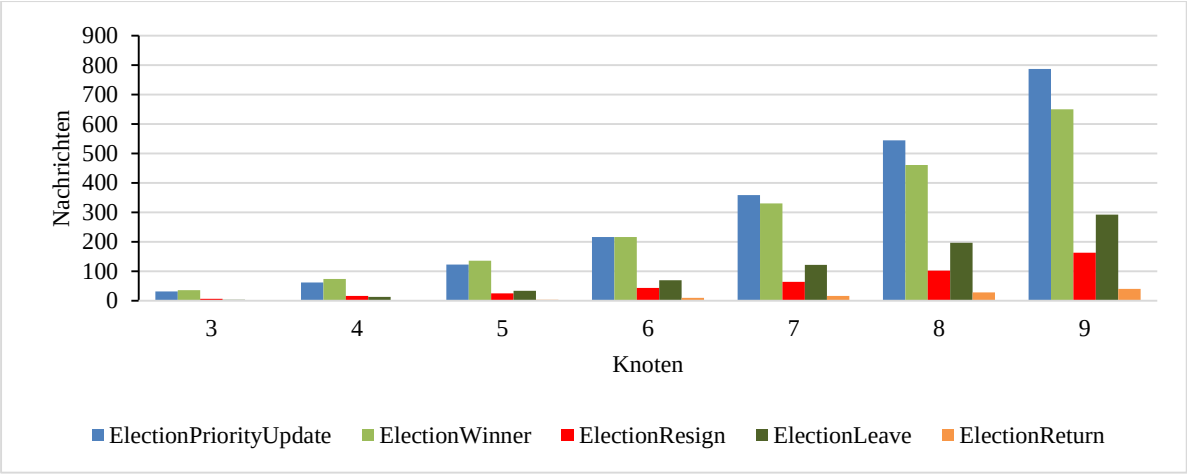

Abbildung 6.20: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Koordinatorenwahl (Masche)

Bei einer reinen Maschentopologie, bei der alle *n* Knoten zueinander direkte Links besitzen, ergibt sich die Anzahl der notwendigen L0-Clustermanager im Netzwerk aus: *n * (n -1)*. Da für jeden Link genau eine Instanz eines L0-Koordinators erstellt wird, ist ihre Gesamtzahl: $\frac{n*(n-1)}{2}$. Beides wird durch die Messwerte aus Abbildung 6.16 und Abbildung 6.17 bestätigt, sie zeigen jeweils einen quadratischen Verlauf für die Anzahl von erzeugten Entitäten auf Hierarchielevel 0. Dies wirkt sich ebenfalls auf die Anzahl der jeweils notwendigen Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl aus, deren quadratisches Wachstum in Abbildung 6.18 dargestellt ist. Zur weiteren Analyse dieser Werte stellen Abbildung 6.19 und Abbildung 6.20 die Anzahl von auftretenden Nachrichten für jeden Typ dar. Es ist ersichtlich, dass vor allem die *PriorityUpdate*- und *Winner*-Nachrichten das resultierende Nachrichtenaufkommen maßgeblich bestimmen. Des Weiteren sind die zusätzlichen *Leave/Return*-Signalisierungen des DCE-Algorithmus zu erkennen.

Sterntopologie

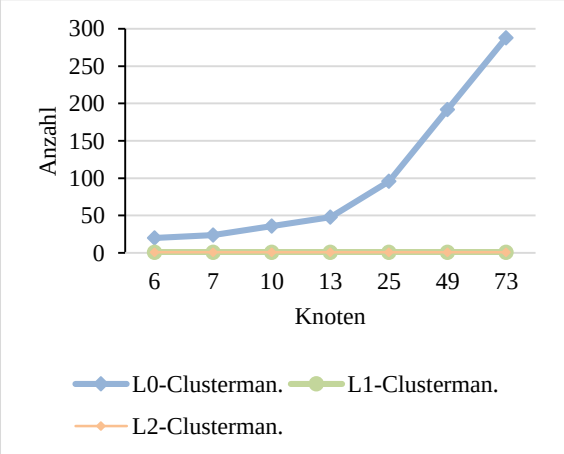

Abbildung 6.21: Clustermanager (Stern)

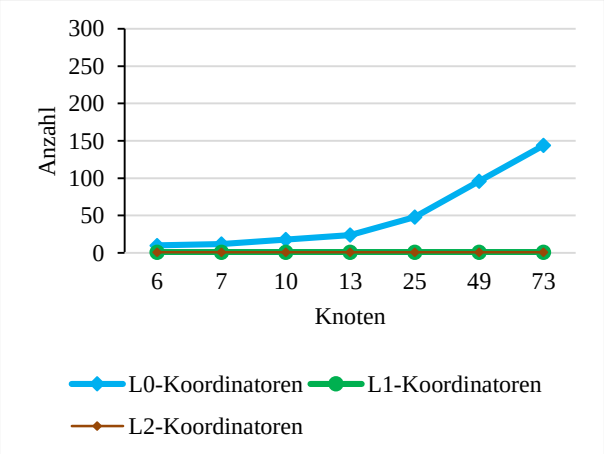

Abbildung 6.22: Koordinatoren (Stern)

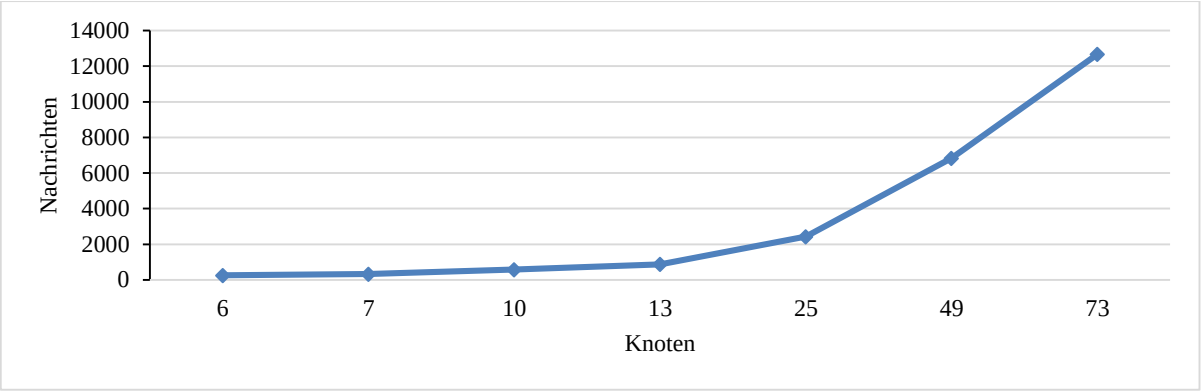

Abbildung 6.23: Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl (Stern)

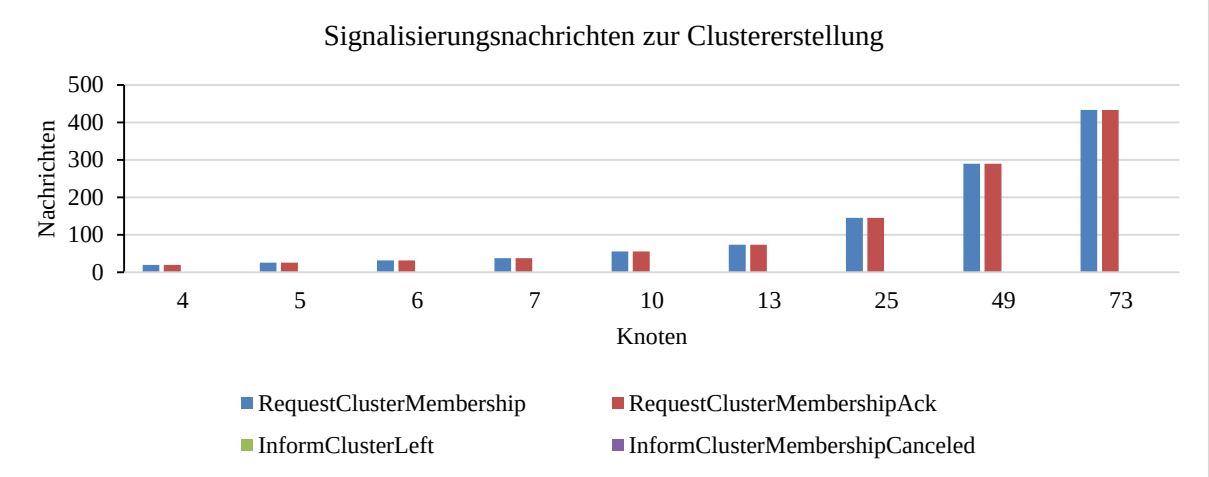

Abbildung 6.24: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Clusterbildung (Stern)

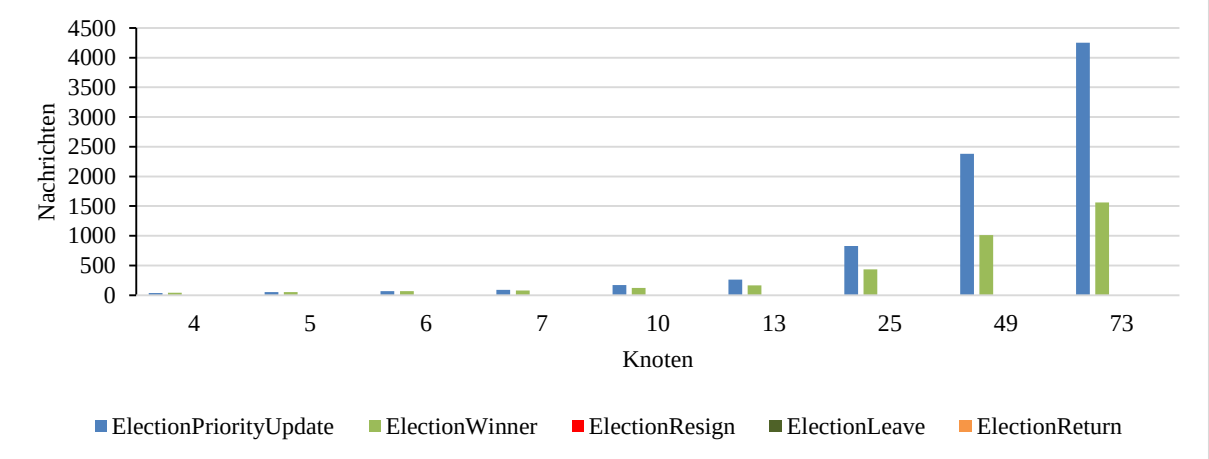

Abbildung 6.25: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Koordinatorenwahl (Stern)

Als zweite Basisstruktur wird die Sterntopologie näher untersucht. In Abbildung 6.21 und Abbildung 6.22 ist zu erkennen, dass bei dieser Topologie die Anzahl der erzeugten Clustermanager und Koordinatoren mit steigender Kontenzahl *n* (*n* beschreibt dabei die Knoten, welche um den zentralen Knoten angeordnet sind) im Gegensatz zur Maschentopologie linear zunehmen. Die Anzahl der L0-Clustermanager und L0-Koordinatoren kann explizit berechnet werden aus: *(4 * n)* bzw. *(2 * n)*. Aus Abbildung 6.23 wird das lineare Wachstum des verursachten Signalisierungsaufkommens für eine steigende Knotenanzahl deutlich. Ähnlich der Maschentopologie besitzen bei den Wahlvorgängen wiederum die *Pri-*

orityUpdate- und *Winner*-Nachrichten den dominierenden Einfluss auf das verursachte Signalisierungsaufkommen. Dies wird anhand der detaillierten Aufstellungen von gemessenen Nachrichten in Abbildung 6.24 und Abbildung 6.25 deutlich.

Broadcast-Domäne

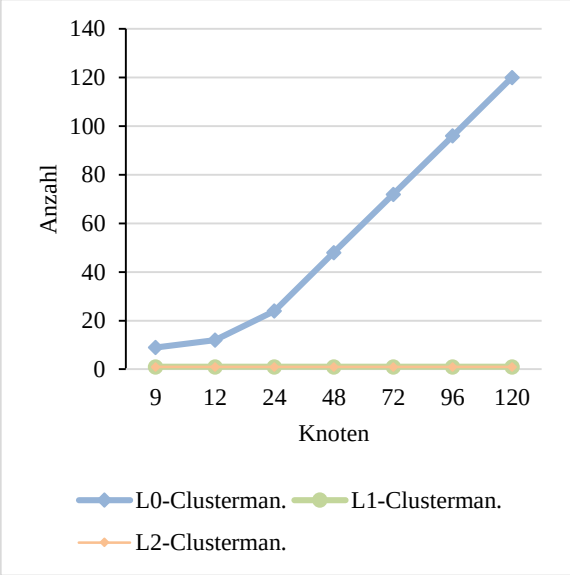

Abbildung 6.26: Clustermanager (Domäne)

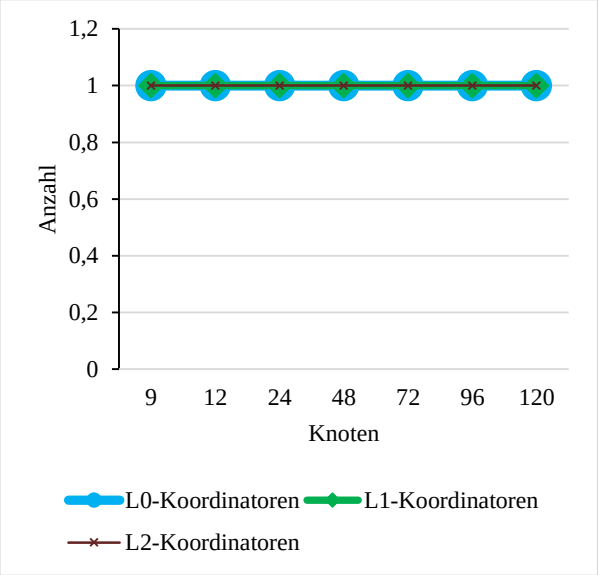

Abbildung 6.27: Koordinatoren (Domäne)

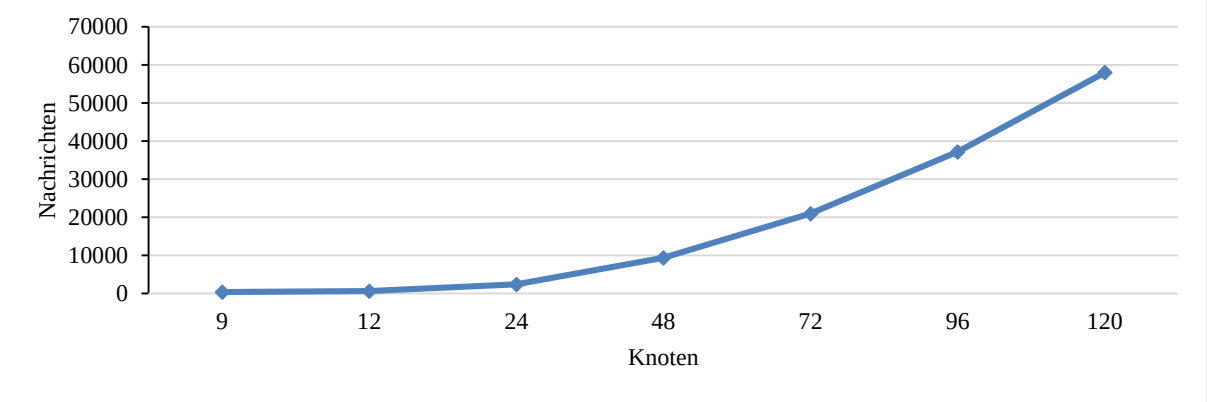

Abbildung 6.28: Signalisierungsnachrichten zur Clusterbildung und Koordinatorenwahl (Domäne)

Abbildung 6.29: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Clusterbildung (Domäne)

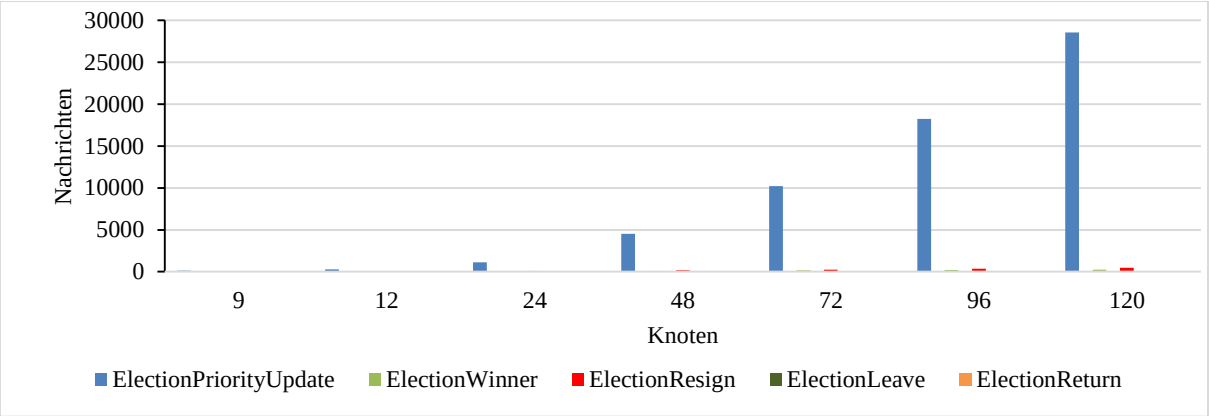

Abbildung 6.30: Detaillierte Verteilung von auftretenden Signalisierungsnachrichten zur Koordinatorenwahl (Domäne)

Betrachtet man größere Broadcast-Domänen (mit mehr als 2 Knoten) erkennt man, dass auf jedem Knoten genau ein L0-Clustermanager für die Wahl des zugehörigen Koordinators erstellt wird. Dies wird durch den linearen Verlauf aus Abbildung 6.26 bestätigt. Anders als zuvor bei der Maschen- und Sterntopologie wird für eine Broadcast-Domäne jedoch trotz steigender Knotenanzahl nur ein einziger L0-Koordinator gewählt. Abbildung 6.27 zeigt die resultierende Koordinatorenanzahl für alle 3 betrachteten Hierarchielevel. Beim Vergleich zwischen beiden Grafiken wird deutlich, dass auf höheren Hierarchielevels jeweils nur ein Clustermanager und der dazu gehörige Koordinator instanziiert werden. Sie befinden sich auf dem Knoten, der den L0-Koordinator für die Domäne gestartet hat[10]. Die für diese Vorgänge notwendigen Signalisierungen sind in Abbildung 6.28 zu sehen. Das abgebildete Nachrichtenaufkommen steigt mit wachsender Knotenanzahl für die Implementierung aus Kapitel 4 quadratisch an. Abbildung 6.29 und Abbildung 6.30 zeigen eine detaillierte Aufstellung anhand der Nachrichtentypen. Daraus wird deutlich, keine *Leave/Return*-Signalisierungen des DCE-Algorithmus auftreten und der Signalisierungsaufwand von der Anzahl von auftretenden *PriorityUpdate*-Nachrichten dominiert[11] wird.

6.2.1.3 Zusammenfassung

Die vorgestellten Experimente haben bestätigt, dass die Signalisierungen der Kontrollebene für alle ausgewählten Basisstrukturen anwendbar sind und die Startphase in endlicher Zeit mit einer finalen Lösung für die Platzierung von notwendigen Managementinstanzen abgeschlossen ist. Die dabei resultierende Struktur der Kontrollebene entsprach jeweils immer den Erwartungen.

	Ring	**Masche**	**Stern**	**Domäne**
Signalisierungsaufkommen	O(n)	O(n²)	O(n)	O(n²)

Tabelle 6.1: Zunahme der Signalisierungen zur Initialisierung in Abhängigkeit von der Knotenanzahl

[10] Falls die Broadcast-Domäne mit anderen Topologiearten kombiniert wird, muss dies nicht zutreffen, da die Platzierung der höheren Koordinatoren immer von der Verteilung der jeweils ungeordneten Koordinatoren abhängt.

[11] Die quadratische Zunahme von *PriorityUpdate*-Signalisierungen kann durch den Einsatz von Broadcast-Nachrichten reduziert werden, sodass diese nicht über die bisher verwendeten Knoten-zu-Knoten-Verbindungen übertragen werden. Bei einer solchen Änderung der Kommunikation muss jedoch beachtet werden, dass eine sichere Zustellung der Aktualisierungsnachrichten nicht mehr gewährleistet ist und es somit zu Inkonsistenzen innerhalb der Kontrollebene kommen könnte. Die dafür notwendigen Untersuchungen sind nicht Bestandteil dieser Arbeit.

Tabelle 6.1 gibt eine Zusammenfassung[12] des ermittelten Verhaltens der Kontrollebene während der Startphase für die ausgewählten Basisstrukturen. Das erwartete Verhalten für die Maschentopologie wird dabei bestätigt: Mit steigender Knotenanzahl steigt das Signalisierungsaufkommen quadratisch an. Ein ähnliches Verhalten wurde für die Platzierung von Managementinstanzen innerhalb einer Broadcast-Domäne ermittelt, die ermittelten Ergebnisse sind jedoch implementierungsspezifisch und Verbesserungen in der Kommunikation sind denkbar, welche nicht Bestandteil dieser Arbeit sind.

Des Weiteren zeigten die Untersuchungen zum Clusterradius, dass der gewählte Wert einen Einfluss auf den Signalisierungsaufwand beim Aufbau der Kontrollebene hat. Eine Anpassung des Clusterradius in Abhängigkeit vom Durchmesser des verwendeten Netzwerks ist sinnvoll (aber für die Funktionsweise von HRM nicht zwingend erforderlich).

6.2.2 Adresszuweisung

Der Übergang von der Initialisierungsphase der Kontrollebene zur Betriebsphase erfolgt mit Hilfe der Adresszuweisung. Entsprechend Abschnitt 3.4 wird dabei jeder Netzwerkschnittstelle eine eindeutige Adresse zugeordnet. Die Bewertung dieser Vorgänge erfolgt durch Zählung der dafür notwendigen *AssignHRMD*-Nachrichten. Der exakte Inhalt einer solchen Signalisierung ist in Anhang B.2 zu finden. Die nachfolgenden Betrachtungen analysieren den Kommunikationsaufwand für eine vollständige Aktualisierung der Adresszuweisungen, sodass dadurch eine *worst-case*-Einschätzung gegeben wird, welche für eine lokal begrenzte Änderung keinesfalls überschritten wird.

6.2.2.1 Einfluss des Clusterradius am Beispiel der Ringtopologie

Ähnlich der Betrachtungen zur Erstellung der Hierarchie wird auch bei der Adresszuweisung zuerst der Einfluss des Clusterradius auf die Anzahl notwendiger Nachrichten untersucht.

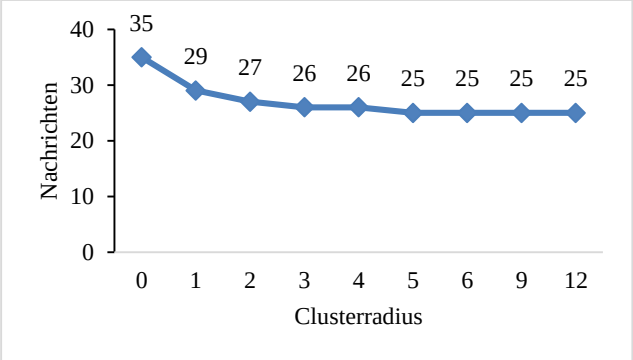

Abbildung 6.31: Signalisierungsnachrichten zur Adresszuweisung (Ring mit 12 Knoten)

Abbildung 6.31 zeigt die resultierende Verteilung in Abhängigkeit vom eingesetzten Radius. Dabei wurden alle Signalisierungen zwischen den Entitäten beachtet, sodass auch knoteninterne Zuweisungen berücksichtigt werden. Die Anzahl der notwendigen Nachrichten ist für Clusterradius 0 am höchsten, während sie in Richtung höherer Radien stetig abnimmt. Des Weiteren ist zu erkennen, dass zwischen den

[12] Die genaue Anzahl der notwendigen Nachrichten für jedes Protokoll kann aufgrund der dabei zu beachtenden Freiheitsgrade nicht ohne Einschränkungen in allgemeiner Form analytisch ermittelt werden. Die dafür notwendige Formel muss insbesondere die Vielzahl von auftretenden nebenläufigen Signalisierungen und die durch das Netzwerk verursachten Verzögerungen beachten. Insbesondere die Permutation der Reihenfolge von empfangenen Nachrichten spielt dabei eine wichtige Rolle. Durch definierte Vereinfachungen kann die Komplexität reduziert werden. Dadurch wird jedoch auch die Aussagekraft der resultierenden Formel stark eingeschränkt, sodass in dieser Arbeit empirische Ergebnisse favorisiert wurden. Zusätzlich geben die ermittelten Werte einen Überblick über die zu erwartenden Größenordnungen der auftretenden Signalisierungen, der bei einfacher qualitativer Bewertung des verursachten Signalisierungsaufwands nicht gegeben wäre.

Werten 5, 6, 9 und 12 keine Unterschiede auftreten. Ursache dafür ist die Instanziierung der Kontrollebene, die für das Szenario für einen Radius *r > 4* stets zur gleichen finalen Lösung führt.

Signalisierung	r = 0	r = 1	r = 2	r = 3	r = 4	r = 5	r = 6	r = 9	r = 12
L0 ➔ L0	12	12	12	12	12	12	12	12	12
L1 ➔ L0	12	12	12	12	12	12	12	12	12
L2 ➔ L1	11	5	3	2	2	1	1	1	1
Gesamt	35	29	27	26	26	25	25	25	25

Tabelle 6.2: Signalisierungen der *AssignHRMID*-Nachrichten zwischen den Hierarchielevels

Zum Verständnis der ermittelten Werte zeigt Tabelle 6.2 die Verteilung der Signalisierungen zwischen den Hierarchielevels. Daraus wird ersichtlich, dass sich ausschließlich die Zeile „L2 ➔ L1“ in Abhängigkeit von der jeweiligen Radiuskonfiguration verändert und die darin beschriebenen Zuweisungen des TOP-Koordinators die Veränderungen an der notwendigen Gesamtmenge verursachen. Insofern der gewählte Radius den Wert 4 überschreitet, sind aufgrund der konstanten Platzierung der Koordinatorinstanzen stets 25 Nachrichten für eine vollständige Adresszuweisung notwendig.

Ähnlich zu Abschnitt 6.2.1.1 erscheint ein Radius *r = 0* eher ungünstig, da sich dadurch im Vergleich zu den besten Lösungen (*r > 4*) eine Steigerung des Signalisierungsaufwands um 40 % einstellt. Jedoch muss bei dieser Betrachtung im Vergleich zur Erstellungsphase der Kontrollebene beachtet werden, dass die verursachten Gesamtkosten der Adresszuweisung auch für eine vollständige Aktualisierung in jedem Fall eher gering ausfallen. Die Signalisierungskomplexität für eine Hierarchietiefe von 3 kann verallgemeinert werden und ergibt sich für die Ringtopologie aus der Summe aus der Anzahl von Netzwerkschnittstellen aller Broadcast-Domänen und der Menge von instanziierten L1-Koordinatoren.

6.2.2.2 Einfluss der Topologie und der Anzahl von Knoten

Als weitere Basisstrukturen stehen die Maschenstruktur, der Stern und die Broadcast-Domäne im Fokus der Untersuchungen.

Ringtopologie

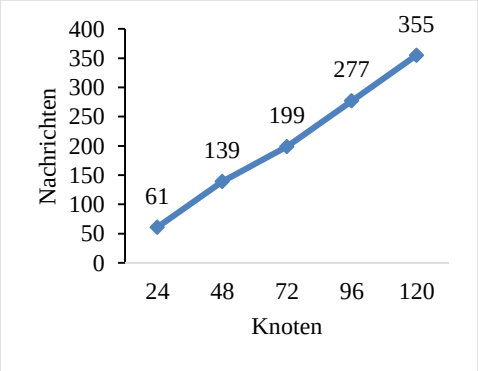

Abbildung 6.32: Signalisierungsnachrichten zur Adresszuweisung (Ring)

Für die Ringtopologie ist in Abbildung 6.32 der Signalisierungsaufwand zu sehen. Daraus ist ein linearer Verlauf mit zunehmender Knotenanzahl abzulesen.

Maschentopologie

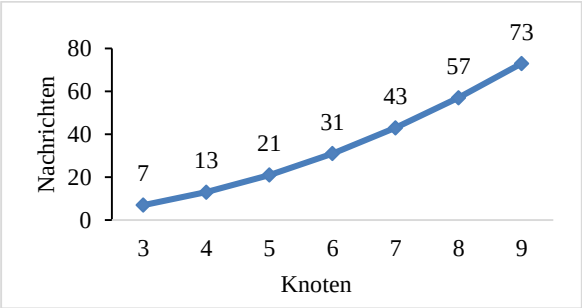

Abbildung 6.33: Signalisierungsnachrichten zur Adresszuweisung (Masche)

Abbildung 6.33 zeigt die Anzahl von gemessenen Adresszuweisungen für die Maschentopologie. Unabhängig von der Knotenanzahl gehört zu den Signalisierungen eine Adresszuweisung an den L1-Koordinator, welche durch den L2-Koordinator ausgelöst wird. Erst dadurch wird die nachfolgende Vergabe von HRMIDs an alle Netzwerkschnittstellen möglich. Die insgesamt notwendige Anzahl von Signalisierungsnachrichten kann in Abhängigkeit von den vorhandenen Netzwerkschnittstellen für *n* Knoten allgemein aus *n * (n - 1) + 1* berechnet werden.

Sterntopologie

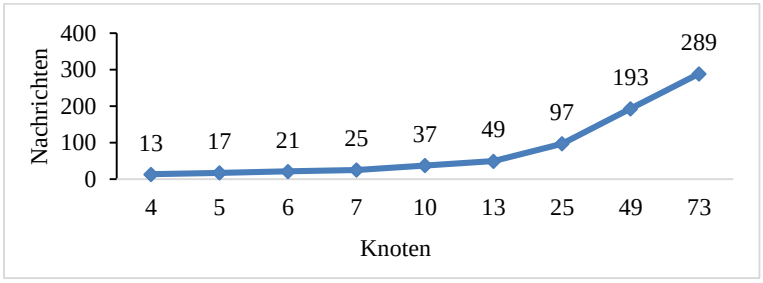

Abbildung 6.34: Signalisierungsnachrichten zur Adresszuweisung (Stern)

Als nächste Basisstruktur wird die Sterntopologie ausgewertet, sie besteht aus jeweils *n* Knoten, welche mit jeweils zwei Links an einen zentralen zusätzlichen Knoten angebunden sind. Somit bedeutet jeder zusätzliche Knoten, dass vier weitere Netzwerkschnittstellen im Netzwerk hinzukommen. Abbildung 6.34 zeigt die Anzahl resultierender Nachrichten in Abhängigkeit von der Anzahl von existierenden Knoten. Das Verhalten der Signalisierungen ist analog der zuvor betrachteten Maschentopologie, sodass sich die Gesamtzahl notwendiger Nachrichten für die Topologie allgemein ergibt aus: *4 * n + 1*. Der erste Summand gibt dabei die Anzahl von Netzwerkschnittstellen und der zweite die Anzahl L1-Koordinatoren an.

Broadcast-Domäne

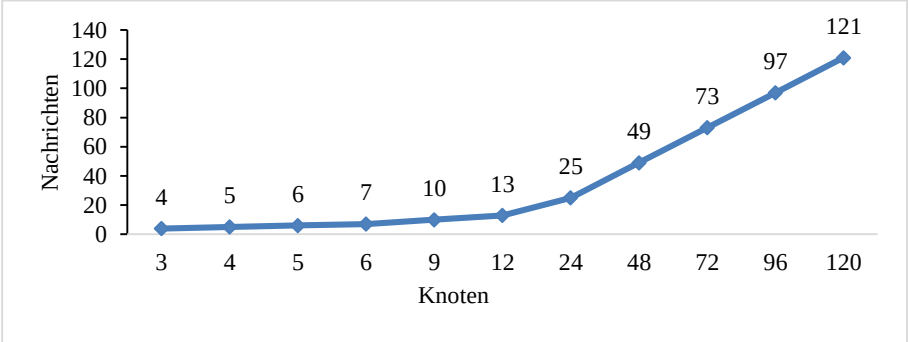

Abbildung 6.35: Signalisierungsnachrichten zur Adresszuweisung (Domäne)

Als letzte Basisstruktur wird die Broadcast-Domäne betrachtet, Abbildung 6.35 zeigt die Anzahl notwendiger Nachrichten für eine steigende Knotenanzahl. Dabei ist zu beachten, dass während der Erstellungsphase der Kontrollebene einer der Knoten als Sieger der Wahlen auf allen Hierarchielevels hervorgeht, sodass alle anderen Knoten keine Koordinatorinstanzen besitzen. Beim Vergleich der resultierenden *AssignHRMID*-Nachrichten fällt die Analogie zu den bisherigen Ergebnissen auf, da sich das Ergebnis wiederum aus der Summe der Anzahl von Netzwerkschnittstellen und der Anzahl L1-Koordinatoren ergibt. Letztere besitzt für dieses Szenario immer den Wert 1.

6.2.2.3 Zusammenfassung

Neben der Wahl des Clusterradius ist die Hierarchietiefe als weiterer Einflussfaktor auf die Signalisierungskomplexität zu nennen. Für größere Werte ergibt sich eine entsprechende größere Anzahl von notwendigen AssignHRMID-Nachrichten, um eine vollständige Aktualisierung der Adresszuweisung im Netzwerk durchzuführen.

$$m = \sum_{n=1}^{k} i_n + \sum_{l=1}^{H-2} c_l$$

Formel 6.1: Anzahl von notwendigen AssignHRMID-Nachrichten für eine vollständige Aktualisierung

Der Signalisierungsaufwand kann allgemein als die Anzahl *m* von notwendigen Nachrichten ausgedrückt und mit Hilfe von Formel 6.1 berechnet werden. Dabei beschreibt der erste Teil die notwendigen Nachrichten, welche für die Adressvergabe an die im Netzwerk vorhandenen Netzwerkschnittstellen benötigt werden. Der zweite Teil beinhaltet die notwendigen Nachrichten, welche für die Adressvergabe an die existierenden Koordinatoren notwendig sind. Die resultierende Formel verwendet folgende Eingabe sind:

- i_n : die Anzahl von Netzwerkschnittstellen des jeweiligen Knotens *n* (wobei k die Anzahl von Knoten im Netzwerk angibt und alle Knoten kontinuierlich mit 1 beginnend nummeriert sind)
- c_l : die Anzahl von Koordinatoren für das Hierarchielevel l (wobei H die Hierarchietiefe angibt und der TOP-Koordinator nicht in die Berechnung einfließt, da er keinen übergeordneten Koordinator besitzt und somit auch keine Adresse zugewiesen bekommt)

Durch den zweiten Summanden wird die Tiefe der Hierarchie in die Berechnung einbezogen, sodass das Ergebnis eine genaue Wiedergabe der Realität darstellen **kann**[13]. Des Weiteren lässt sich aus Formel 6.1 eine lineare Kommunikationskomplexität *O(i) + O(c)* erkennen, welche von der Anzahl an Netzwerkschnittstellen *i* und Koordinatoren *c* im Netzwerk abhängig ist. Dies wirkt sich am stärksten bei einer Maschentopologie auf die Anzahl notwendiger Signalisierungsnachrichten aus.

Im Vergleich zu Abschnitt 6.2.1 ist der Einfluss der Signalisierungen zur Adressvergabe auf die resultierende Gesamtanzahl von Nachrichten während der Startphase jedoch gering, sodass sie bei der Bewertung der Kosten von HRM beim Netzwerkstart im Hintergrund stehen.

[13] Einer Ermittlung von exakten Ergebnissen steht die in der Formel einzusetzende Anzahl von Koordinatoren für die unterschiedlichen Hierarchielevels gegenüber. Ist die zu erwartende Verteilung der Koordinatorinstanzen nicht aufgrund der Topologie offensichtlich, muss die resultierende Verteilung empirisch ermittelt werden. Dieses Vorgehen wurde in dieser Arbeit auf die Basistopologien angewandt und die notwendigen Nachrichten ebenfalls empirisch bestimmt.

6.3 Signalisierungs- und Speicheraufwand der Kontrollebene in der Betriebsphase

Zur Bewertung der Betriebsphase sind die Kosten interessant, welche durch die periodisch wiederkehrenden Signalisierungen der Protokolle zur Aktualisierung der Managementinfrastruktur und zur Verteilung von Routingdaten verursacht werden. Ebenso muss der Speicheraufwand für einzelne Knoten näher betrachtet werden. Er wird maßgeblich durch die Kommunikations- und Routingdaten bestimmt.

6.3.1 Signalisierungsaufwand

Die periodischen Signalisierungen der Kontrollebene bestehen aus folgenden Nachrichten:

- *AnnounceNeighborNode* (kontinuierliche Nachbarschaftserkennung, siehe Abschnitt 3.3.1.1)
- *AnnounceCoordinator* (Bekanntgabe von Koordinatoren, siehe Abschnitt 3.3.2.4)
- *RouteReport* (Signalisierung von Routingdaten aufwärts der Hierarchie, siehe Abschnitt 3.5.2)
- *RouteShare* (Signalisierung von Routingdaten abwärts der Hierarche, siehe Abschnitt 3.5.3)

Bei den nachfolgend vorgestellten Untersuchungen wurde in der Implementierung für jeden der aufgeführten Nachrichtentypen ein Beispiel einer bitgenauen Kodierung aller enthaltenen Informationen als Grundlage der Messungen verwendet[14]. Über einen festgelegten Zeitraum von 30 Minuten wurde für jeden Link das Auftreten der einzelnen Nachrichtentypen gezählt. Anschließend wurde daraus in Kombination mit der festgelegten Definition für die Kodierung der einzelnen Nachrichtentypen das verursachte durchschnittliche Datenaufkommen für den betrachteten Zeitraum in Bezug auf Schicht 3 des OSI-Modells bestimmt. Bei allen Messungen wurde dabei die Konnektivität im Netzwerk unverändert gelassen, sodass Umstrukturierungen der Kontrollebene und eine dadurch verursachte Beeinflussung der Signalisierungsrate ausgeschlossen wurden.

Zur Ermittlung von aussagekräftigen Ergebnissen werden ausschließlich die beiden Extremfälle untersucht: die minimal und die maximal zu erwartenden Signalisierungskosten. Durch diese Betrachtung wird deutlich, in welchem Wertebereich sich die durch die Signalisierungen verursachten Kosten bewegen:

1. **Minimale Signalisierungskosten (konstante QoS-Eigenschaften):** Um eine Einschätzung des minimal verursachten Datenaufkommens zu ermöglichen, werden die Eigenschaften konstant gehalten. Folglich treten keine Veränderungen an den Routingdaten auf und die *RouteReport/RouteShare*-Signalisierungen bestehen ausschließlich aus den periodischen Vollaktualisierungen, welche entsprechend des Sendeintervalls T_{voll} gesendet werden (siehe Abschnitt 3.5.4.2).
2. **Maximale Signalisierungskosten (stetig variierende QoS-Eigenschaften):** Dadurch wird das maximal verursachte Datenaufkommen untersucht. Dieser Fall tritt ein, wenn jede signalisierende Entität der Kontrollebene stets alle Routen als „verändert“ annimmt. Innerhalb der Implementierung wird dies durch Deaktivierung der automatischen Datenreduktion innerhalb der Routingdatenverteilung erreicht. Die Menge der jeweils signalisierten Daten der *Report/RouteShare*-Nachrichten entspricht somit Vollaktualisierungen mit konstanter Größe und dem Intervall T_{min} (siehe Abschnitt 3.5.4.2). Durch diese Messungen wird eine Obergrenze für die durch Signalisierungen verwendeten Netzwerkressourcen für die ausgewählten Topologien gegeben.

Bei den nachfolgenden Messungen werden folgende Annahmen für die Signalisierungen verwendet:

- *AnnounceCoordinator*-Nachrichten

[14] Anhang B gibt eine Übersicht über die Typen und die jeweils beinhalteten Elemente. Die verwendeten Datengrößen sind der Implementierung zu entnehmen.

Sendeintervall für Koordinatorbekanntgaben: Die Messungen zur Betriebsphase gehen von einem stabilen Netzwerk aus, in dem Veränderungen in der Konnektivität nicht auftreten. Die Hierarchie der Kontrollebene gilt zudem als bereits aufgebaut und wird als stabil angenommen. Daher werden bei den Untersuchungen generell 60 Sekunden als Sendeintervall verwendet. Dies entspricht den Annahmen von Abschnitt 6.2.1.

- *RouteReport*/*RouteShare*-Nachrichten

  Sendeintervall T_{min} für Teilaktualisierungen: Um möglichst aktuelle Routingdaten (siehe Abschnitt 3.10.6) auf allen Knoten zu speichern, wird ein äußerst kurzes Intervall von 1 Sekunde verwendet.

  Sendeintervall T_{voll} für Vollaktualisierungen: Im Vergleich zu den bei OSPF üblichen 30 Minuten wurde an dieser Stelle ein eher kleiner Wert von 3 Minuten gewählt, um möglichst konsistente Daten sicherzustellen.

Es ist zu erwarten, dass die Nachrichten zur Verteilung von Routingdaten bei sehr hoher Netzdynamik den größeren Einfluss auf die Gesamtkosten verursachen, während die Menge der *AnnounceCoordinator*-Nachrichten bei den minimalen Kosten den entscheidenden Einfluss ausüben. Des Weiteren sollte sich der optimale Wert für den Clusterradius zwischen 0 und dem Durchmesser des jeweiligen Netzwerks befinden. Durch Anhebung der Hierarchietiefe sind weitere Kostenreduktionen zu erwarten. Die pro Knoten notwendigen Verbindungen für die Signalisierungen der Kontrollebene sollten proportional zur Anzahl vorhandener Knoten im Netzwerk ansteigen. Ein ähnliches Wachstum ist für die durchschnittliche Größe des pro Knoten notwendigen HRGs zu erwarten.

6.3.1.1 Nachbarschaftserkennung

Die Nachbarschaftserkennung gehört zu den implementierungsspezifischen Signalisierungen des HRM-Konzeptes. Sie unterscheidet sich grundlegend zwischen IP- und FoG-basierten Netzwerken. Da entsprechend Abschnitt 4.2.1 innerhalb der Implementierung die FoG-spezifische Nachbarschaftserkennung verwendet wird, können die bei einer IP-basierten Implementierung verursachten Kosten nicht empirisch bestimmt werden. Stattdessen kann für IP-basierte Netzwerke das durch *AnnounceNeighborNode*-Nachrichten verursachte Signalisierungsaufkommen sehr einfach berechnet werden.

$$d_{AnnounceNeighborNode} = SIZE_{AnnounceNeighborNode} * rate_{AnnounceNeighborNode} * (1 + n)$$

Formel 6.2: Signalisierungskosten (Datenrate) von *AnnounceNeighborNode*-Anfragen eines Knotens

Formel 6.2 zeigt, wie die Datenrate $d_{AnnounceNeighborNode}$ der durch die Anfragen eines Knotens verursachten Signalisierungen ermittelt wird. Dabei werden folgende Parameter verwendet:

- $SIZE_{AnnounceNeighborNode}$: Die Größe einer *AnnounceNeighborNode*-Nachricht hat einen konstanten Wert und hängt von der verwendeten Implementierung ab.
- $rate_{AnnounceNeighborNode}$: Dadurch wird die Rate der Anfragen zur Nachbarschaftserkennung in die Berechnung einbezogen.
- *n :* Entsprechend Abschnitt 3.3.1 ergeben sich die für jede Anfrage durch die Broadcast-Domäne übertragenen Nachrichten aus *(1 + n)*, wobei n die Anzahl von Knoten angibt, welche auf die versandte Anfrage eine Antwort senden.

Für diese Parameter sind folgende Werte sinnvoll:

- $SIZE_{AnnounceNeighborNode}$ = 17 Bytes: Entsprechend Anhang B wird innerhalb einer *AnnounceNeighborNode*-Nachricht eine Knoten-ID des Senders übertragen. Dafür können beispielsweise 16 Bytes zur Kodierung einer UUID verwendet werden. Des Weiteren muss eine

Anfrage-ID sowie ein Marker zur Unterscheidung zwischen Anfrage und Antwort übertragen werden. Beides kann innerhalb eines zusätzlichen Bytes kodiert sein.

- $rate_{AnnounceNeighborNode}$ = 1 Sekunde: Für eine schnelle Reaktion auf ausgefallene Nachbarknoten ist dieser (eher niedrige) Wert sinnvoll.

Somit ergibt sich für einen Link zwischen zwei Knoten nach Formel 6.2 eine Datenrate von *17 * 1 * (1 + 1) = 34 Bytes/s* für einen Knoten. Da beide Nachbarknoten des Links die Signalisierung verwenden, ergibt sich eine resultierende Gesamtdatenrate von *68 Bytes/s* für die auftretenden *AnnounceNeighborNode*-Nachrichten auf jedem Link zwischen zwei Knoten. Dieser Wert tritt unabhängig vom gewählten Clusterradius sowie der aktuell verwendeten Hierarchietiefe für jeden direkten Link im Netzwerk auf. Er wird ausschließlich durch die verwendete Signalisierungsrate beeinflusst. Wird die Rate geringer gewählt (das Signalisierungsintervall steigt somit), fällt der resultierende Wert geringer als 68 Bytes/s aus. Aufgrund seines geringen Maßes und seiner Unabhängigkeit von typischen HRM-Eingabeparametern (Clusterradius und Hierarchietiefe) weist er einen äußerst geringen Einfluss auf die Gesamtbelastung des Netzwerks beim Einsatz der Kontrollebene von HRM auf. Folglich kann er beim Vergleich des Signalisierungsaufkommens für die Ring-, Maschen- und Sterntopologie ohne Verfälschung des Ergebnisses vernachlässigt werden.

Anders verhält es sich bei Broadcast-Domänen. Nimmt man eine Knotenanzahl von 120 für eine solche Domäne an, erhält man unter Annahme der zuvor festgelegten Parameterwerte für jeden Knoten eine verursachte Signalisierungsrate von *17 * 1 * (1 + 120) = 2057 Bytes/s*. Somit beträgt die Gesamtbelastung der Domäne aufgrund von *AnnounceNeighborNode*-Nachrichten in diesem Fall *246840 Bytes/s*. Aus Formel 6.2 lässt sich dabei ein lineares Wachstum der verursachten Datenrate in Abhängigkeit von der Anzahl an Knoten innerhalb der Domäne ablesen[15].

6.3.1.2 Einfluss des Clusterradius am Beispiel der Ringtopologie

Ähnlich der Startphase steht zur Bewertung der Betriebsphase zuerst der Clusterradius als wichtige Eingabegröße im Fokus der Untersuchungen. Dabei wird der bereits bekannte Ring aus 12 Knoten und Links als Beispielnetzwerk verwendet. Er stellt eine kostengünstige Variante zur Bereitstellung von redundanten Links dar und wird somit als eine typische Topologie für Core-Netzwerke ausgewählt[16].

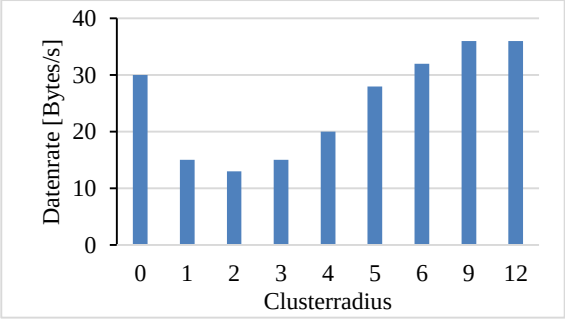

Abbildung 6.36: Minimale Signalisierungskosten (Ring mit 12 Knoten)

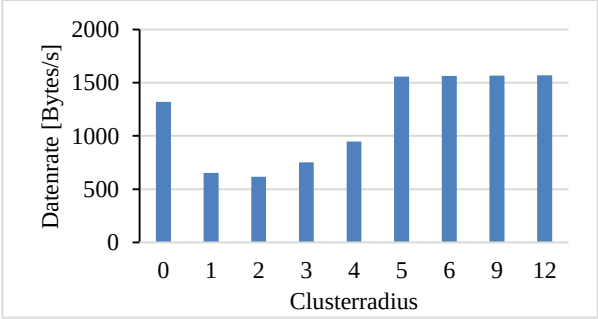

Abbildung 6.37: Maximale Signalisierungskosten (Ring mit 12 Knoten)

[15] Die Implementierung der Nachbarschaftserkennung kann die Anzahl von ausgehenden Anfragen jedes Knotens durch geeignete Auswertung von parallel eintreffenden Anfragen anderer Knoten reduzieren, sodass die Kosten nicht linear mit der Anzahl von Knoten einer Broadcast-Domäne steigen sondern geringer ausfallen. Dies bedarf jedoch weiterer Untersuchungen, welche nicht Bestandteil dieser Arbeit sind.

[16] An dieser Stelle werden die Untersuchungen auf ein Szenario beschränkt, da sonst die Vielzahl von möglichen Szenarien den Rahmen dieser Arbeit sprengen würde. Die Wahl der Ringtopologie erfolgte dabei mit Hinblick auf eine möglichst wahrscheinliche Topologie, bei der HRM auch seine Vorteile durch Einbeziehung von redundanten Routen ausspielen kann.

In Abbildung 6.36 und Abbildung 6.37 sind die durchschnittlichen Belastungen der Netzwerklinks für beide Messtypen zu sehen[17]. Dabei fallen drei Punkte auf:

- Ein Clusterradius von 2 erscheint als optimaler Wert, um möglichst geringe Kosten durch die Verwendung von HRM zu erhalten. Sowohl für einen Clusterradius von 0 als auch bei Verwendung größerer Clusterradien ergeben sich erhöhte Kosten, welche im Vergleich zu Clusterradius 2 für den Minimalfall (linkes Bild) um Faktor 2,8 und im Maximalfall um Faktor 2,5 ansteigen.
- Bei genauerem Vergleich zwischen beiden Abbildungen fällt der unterschiedliche Verlauf für die Radien 5 und 6 auf. Während sich die verursachten Kosten im Minimalfall deutlich von denen von Radius 9 und 12 unterscheiden, ist dies im Maximalfall nicht zu erkennen. Die Ursache ist die durch *AnnounceCoordinator*-Nachrichten verursachte Datenrate und der sich daraus ergebende Einfluss auf das Gesamtergebnis. Im ersten Fall ist dieser im Vergleich zu den anderen Nachrichtentypen eher hoch, während im zweiten Fall die verwendeten *RouteReport/RouteShare*-Nachrichten die signifikant höheren Kosten verursachen, welche sich für alle Radien größer 4 nicht signifikant voneinander unterscheiden.
- Des Weiteren fallen die Werte für die Radien 9 und 12 auf, welche sich in beiden Abbildungen nicht voneinander unterscheiden. Dies hat zwei Ursachen:
    1. Einerseits wird dies durch die implementierte Art der Verarbeitung von *AnnounceCoordinator*-Nachrichten verursacht. Sie leitet die Nachrichten nicht weiter, wenn die darin aufgezeichnete Route zum sendenden Koordinator bereits länger als die kürzeste, bekannte ist. Dadurch wird die Weiterleitung der Nachrichten für diese Topologie bereits ab einer Routenlänge von 6 abgebrochen, sodass sich die Wege für die Radien 9 und 12 nicht voneinander unterscheiden.
    2. Anderseits unterscheidet sich die finale Lösung der Strukturierung der Kontrollebene für beide Radien nicht voneinander, sodass sowohl die Kommunikationswege als auch die auf ihnen signalisierten Datenmengen gleich ausfallen.

Die Messungen zeigen für unterschiedliche Clusterradien erkennbare Unterschiede, wobei für eine hohe Nutzungsdynamik der Einfluss von *AnnounceCoordinator*-Nachrichten eher gering einzuschätzen ist. Stattdessen tragen die übertragenen *RouteReport*/*RouteShare*-Nachrichten maßgebend zur Menge des resultierenden Datenaufkommens bei. Ein ähnlicher Kurvenverlauf ist in Anhang C für eine Ringtopologie aus 48 Knoten dargestellt. Vergleicht man die Werte für beide Szenarien miteinander, wird deutlich, dass für Netzwerke mit größerem Durchmesser[18] ein höherer Clusterradius für möglichst optimale Kosten verwendet werden muss. Dies entspricht der intuitiven Erwartung bei Betrachtung der verwendeten Hierarchie innerhalb der Kontrollebene. Für das Netzwerk aus Anhang C erscheint für eine Hierarchietiefe von 3 ein Wert von 6 optimal in Bezug auf die dadurch verursachten Kosten. Unabhängig vom letztlich gewählten Clusterradius liegen die dabei resultierenden Datenraten der Signalisierungen für die maximal zu erwartende Dynamik der Topologie im Bereich von wenigen Kilobytes pro Sekunde. Diese Werte erscheinen in Bezug auf die maximal erreichbaren Datenraten von heute typischen Links (mit einer Übertragungsrate von 1 oder 10GBit/s) als akzeptabel und können durchaus auf einige 100KB/s ansteigen, ohne den Netzwerkverkehr signifikant zu beeinträchtigen.

[17] Die im nachfolgenden Abschnitt 6.4.2 diskutierten Routen durch fremde Cluster wurden während den Messungen ebenfalls signalisiert, sodass das Szenario den Fall betrachtet, bei dem HRM seine Vorteile unabhängig des Clusterradius bietet.

[18] Als Durchmesser wird das Maximum der kürzesten Routen zwischen zwei beliebigen Knoten eines Netzwerks verstanden.

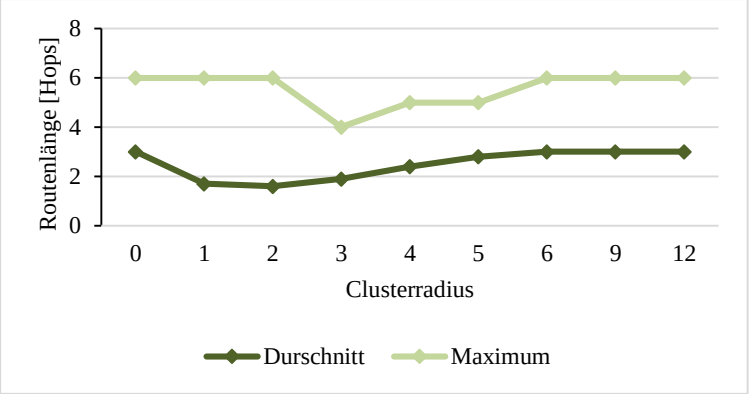

Abbildung 6.38: Routenlänge zwischen den Entitäten der Kontrollebene (Ring mit 12 Knoten)

Nachdem zuvor die Unterschiede zwischen minimal und maximal zu erwartenden Kosten untersucht wurden, verbleibt die Frage, wodurch die zuvor ermittelten Kurvenverläufe beeinflusst werden. Als mögliche Ursache kommt dabei die Länge der Übertragungswege zwischen den Entitäten der Kontrollebene in Frage. Sie sollte für sehr kleine und sehr große Radien aufgrund der resultierenden Struktur der Hierarchie und den damit verbunden extremen Clustergrößen erhöht ausfallen. Abbildung 6.38 stellt zur Untersuchung dieses Zusammenhanges die ermittelten Längen für das Beispielnetzwerk dar. Für einen Clusterradius von 0 sowie für Werte oberhalb von 5 lässt sich erkennen, dass die durchschnittliche Länge der Routen im Vergleich zu den anderen Radien höher ausfällt und somit die Signalisierungsdaten mehr Knoten passieren müssen, was wiederum die resultierende Datenrate der Signalisierungen negativ beeinflusst. Somit entspricht der Kurvenverlauf der Erwartung und zeigt, dass durch geeignete Wahl des Clusterradius die Routenlänge zwischen den Entitäten der Kontrollebene positiv beeinflusst werden kann. Die Wahl des Clusterradius sollte in Hinblick auf die Skalierbarkeit der Signalisierungen für sehr große Netzwerke getroffen werden. Anhang C zeigt für eine Hierarchietiefe von 3, dass bei vierfacher Netzwerkgröße statt einem Wert von 2 eher ein Clusterradius von 6 als optimale Wahl erscheint.

6.3.1.3 Einfluss der Topologie und der Anzahl von Knoten

Nachdem die Ringtopologie in den vorherigen Abschnitten bereits mit variierendem Clusterradius untersucht wurde, wird nachfolgend das Verhalten der periodischen Signalisierungen der Kontrollebene für alle ausgewählten Basisstrukturen näher betrachtet.

Ringtopologie

Im Folgenden wird eine Ringtopologie verwendet, dabei variiert jedoch die Anzahl von Knoten zwischen 24 und 168. Dabei wurden eine Hierarchietiefe von 3 und ein Clusterradius von 8 (wie zuvor in Abschnitt 6.2.1.2) verwendet, sodass dadurch das letztlich verursachte Signalisierungsaufkommen im Netzwerk eher gering ausfällt[19]. Durch die Wahl dieser Hierarchietiefe wird ebenfalls die maximal zu erwartende Verzögerung bis zur Aktualisierung von entfernten Routingdaten klein gehalten (siehe Abschnitt 3.10.6).

[19] Weitere Details zum Vergleich zwischen Hierarchietiefe 3 und 4 werden in Abschnitt 6.3.1.4 gegeben.

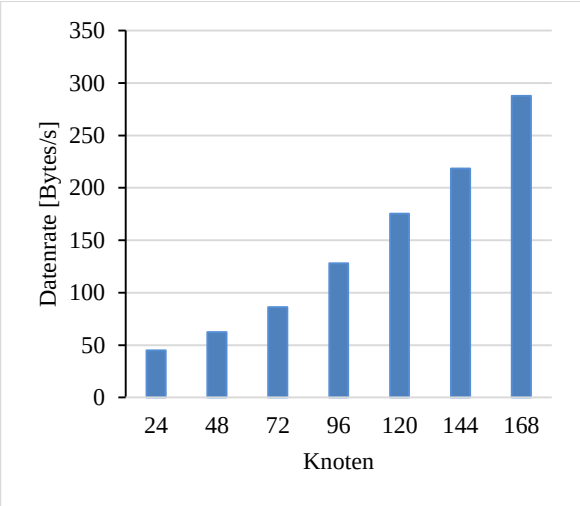

Abbildung 6.39: Minimale Signalisierungskosten mit Hierarchietiefe 3 (Ring)

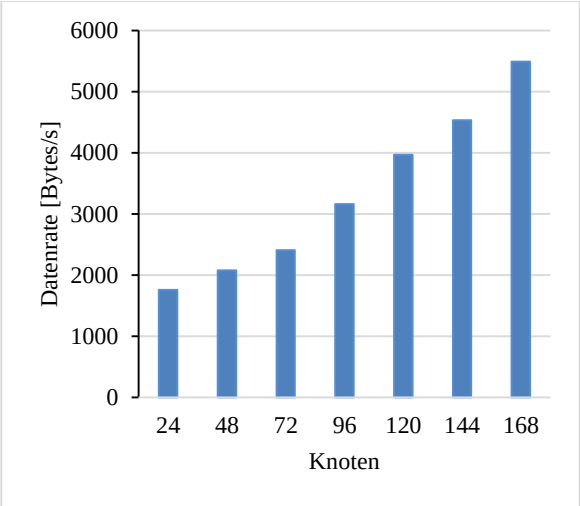

Abbildung 6.40: Maximale Signalisierungskosten mit Hierarchietiefe 3 (Ring)

In Abbildung 6.39 und Abbildung 6.40 wird der lineare Anstieg der Kosten deutlich, wobei die Kosten für eine Knotenanzahl größer 72 schneller ansteigen, während sie für eine kleinere Knotenanzahl vergleichsweise langsamer zunehmen. Dieser Unterschied wird nachfolgend nochmals näher untersucht.

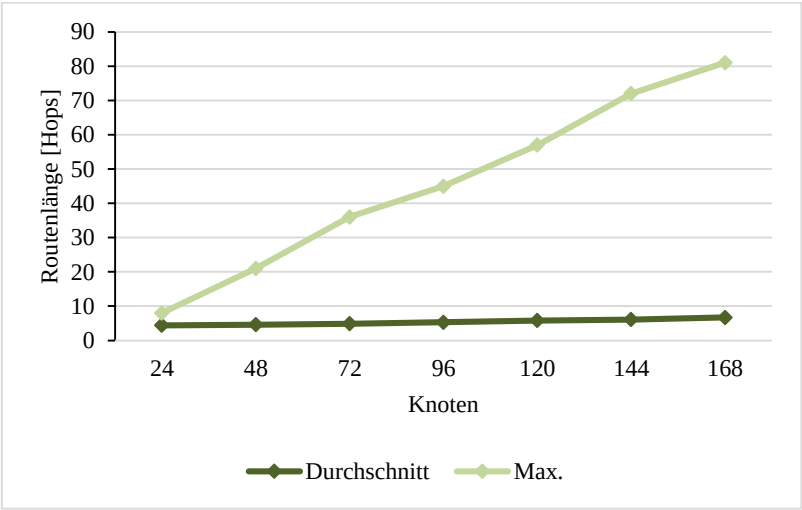

Abbildung 6.41: Routenlänge zwischen den Entitäten der Kontrollebene (Ring)

Auch für die in Abbildung 6.41 dargestellte Routenlänge in Abhängigkeit von der Knotenanzahl lässt sich ein linearer Anstieg erkennen, der jedoch geringfügige Abweichungen besitzt. Sie werden durch die Platzierung der Koordinatorinstanzen verursacht, welche abhängig von der jeweiligen Topologie ist. Dadurch entspricht die maximal auftretende Routenlänge nicht immer dem Durchmesser des jeweiligen Netzwerks und die Belastung des Netzwerks mit Signalisierungsdaten variiert.

Maschentopologie

Die Maschentopologie besteht aus Knoten, welche alle zueinander paarweise einen direkten Link besitzen. Dadurch ergibt sich eine gute Absicherung gegenüber Ausfällen einzelner Links. Nachteilig dabei ist jedoch die hohe Konnektivität, welche sich negativ auf die Signalisierungskosten auswirkt.

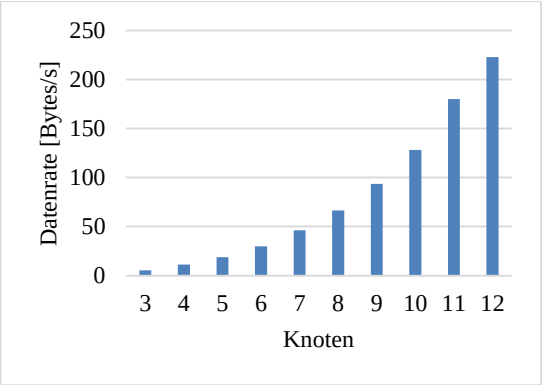

Abbildung 6.42: Minimale Signalisierungsdaten mit Hierarchietiefe 3 (Masche)

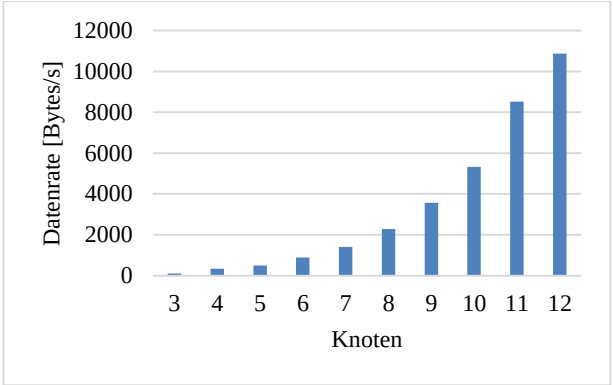

Abbildung 6.43: Maximale Signalisierungsdaten mit Hierarchietiefe 3 (Masche)

Sowohl in Abbildung 6.42 als auch in Abbildung 6.43 ist der nichtlineare Verlauf der Kosten bei einem Clusterradius von 8 in Abhängigkeit von der Anzahl an Knoten erkennbar. Der rasche Anstieg erscheint quadratisch und wird hauptsächlich durch die aufkommenden *RouteShare*-Nachrichten verursacht. Sie beschreiben die Alternativrouten zu allen Zielen, deren Menge durch die quadratisch steigende Anzahl von Links maßgeblich beeinflusst wird. Die kürzeste Route zwischen den Entitäten der Kontrollebene hat dabei immer eine Länge von 1.

Sterntopologie

Da die in dieser Arbeit betrachte Form der Sterntopologie eines Core-Netzwerks aus einem zentralen Router und den angebundenen Gateway-Routern besteht, kann nur der zentrale Router zusätzliche Routen von den umliegenden Gateways signalisiert bekommen. Dabei erkennt die HRM-Implementierung, welche Routen aufgrund der Nachbarschaftserkennung bereits auf dem zentralen Router bekannt sein müssen und verhindert deren Übermittlung in *RouteReport/RouteShare*-Nachrichten. Da das bei dieser Topologie alle Routen eines Gateway-Routers betrifft, werden mit steigender Knotenanzahl keine zusätzlichen *RouteReport*-Nachrichten versendet. Analog dazu verhalten sich aufgrund der gleichen Optimierung die *RouteShare*-Signalisierungen. Somit steigen die resultierenden Signalisierungskosten ausschließlich in Abhängigkeit von der Anzahl der Koordinatorinstanzen auf dem zentralen Knoten an – es besteht eine lineare Abhängigkeit, wobei die Signalisierungen ausschließlich lokal erfolgen.

Broadcast-Domäne

Ohne explizite Messungen ist das Signalisierungsverhalten einer abgeschlossenen Broadcast-Domäne (ohne Verbindung zu anderen Netzwerken) bestimmbar. Wie in Abschnitt 6.2.1.2 beschrieben, gibt es in diesem Fall nur einen Knoten, der alle Koordinatorinstanzen beinhaltet, und die anderen Mitglieder der Domäne stellen die sogenannten Endsysteme gegenüber diesen Instanzen dar. Entsprechend Abschnitt 3.3.5.3 erhalten sie somit keine *AnnounceCoordinator*-Nachrichten. Das Gleiche gilt für *RouteReport*/*RouteShare*-Nachrichten, da sich in einer abgeschlossenen Broadcast-Domäne prinzipiell alle direkten Nachbarn kennen und kein Austausch von Routingdaten notwendig ist. Somit entsteht kein Signalisierungsaufwand innerhalb einer abgeschlossenen Broadcast-Domäne.

6.3.1.4 Einfluss der Clusterunterteilung und der Zielaggregation

Aufgrund der in den Signalisierungen zur Verteilung von Routingdaten angewandten Zielaggregation ergeben sich bei komplexeren Netzwerken Vorteile für das resultierende Signalisierungsaufkommen.

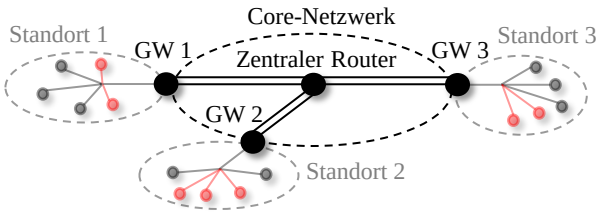

Abbildung 6.44: Sterntopologie mit hinzukommenden Endknoten

Abbildung 6.44 stellt die Sterntopologie aus Abschnitt 6.1.1 dar, wobei zusätzlich die rot markierten Endknoten zur Topologie hinzugefügt worden sind. Vergleicht man das durch die Signalisierungen verursachte Datenaufkommen mit dem der ursprünglichen Netzwerkkonfiguration, stellt man für die Verteilung von *RouteReport/RouteShare*-Nachrichten fest:

- Das Core-Netzwerk wird auch zusätzlichen Endknoten nicht durch zusätzliche Signalisierungsdaten belastet.
- Die Broadcast-Domäne jedes Standortes wird zusätzlich belastet:
    - Jeder zusätzliche Endknoten (rot dargestellt) muss ebenfalls Routingdaten zu den entfernten Netzwerkabschnitten signalisiert bekommen.
    - Jedoch bleibt die Menge der an die ursprünglichen Endnoten (grau dargestellt) übertragenen Routingdaten unverändert.

Das am Beispiel der Ringtopologie gezeigte Verhalten zum Anstieg des Signalisierungsaufwands kann für verschiedenste andere komplexere Szenarien gezeigt werden. Es wäre dabei auch denkbar, dass im Core-Netzwerk aus Abbildung 6.44 zusätzliche Knoten eingefügt werden. Der letztlich durch den Einsatz von Clustern und der damit verbundenen Zielaggregation erzielte Vorteil (in Bezug auf eine Signalisierung ohne Topologieaggregation) ist immer individuell für die jeweils verwendete Topologie, sodass eine allgemeingültige quantitative Aussage nicht möglich ist.

6.3.1.5 Einfluss der Hierarchietiefe

Der gewählte Wert für die Tiefe der Hierarchie ist ein weiter wichtiger Einflussfaktor auf die Kosten der Kontrollebene. Für eine Tiefe von 2 wird für jede Broadcast-Domäne jeweils ein L0-Koordinator platziert. Das resultierende Hierarchielevel 0 wird dabei durch einen globalen TOP-Koordinator auf Hierarchielevel 1 verwaltet, welcher für alle untergeordneten L0-Koordinatoren sowohl Adressen als auch Routingdaten verwaltet. Der gewählte Clusterradius hat somit in diesem Fall keinen Einfluss auf die Strukturierung der Kontrollebene und der DCE-Algorithmus kommt nicht zur Anwendung. Die Struktur einer solchen Kontrollebene ähnelt der einer *OSPF Area* mit instanziiertem *Designated Router*, der als zentraler Ankerpunkt der Signalisierungen dient. In dieser Arbeit wird stattdessen eine Hierarchietiefe größer 2 favorisiert, da dadurch eine gute Datenreduktion und somit auch eine gute Skalierbarkeit für größere Netzwerke gewährleistet wird.

Größere Hierarchietiefen führen zu zusätzlichen Hierarchielevels und somit auch zu weiteren Koordinatorinstanzen innerhalb der Kontrollebene. Durch ihre *AnnounceCoordinator*-Nachrichten ist ein Anstieg der Datenrate von Signalisierungen zu erwarten. Interessant ist dabei die Frage, inwiefern Veränderungen der Hierarchietiefe zu signifikanten Sprüngen der Kosten von HRM führen. Des Weiteren ist es denkbar, dass eine veränderte Hierarchietiefe auch den optimalen Wert für den Clusterradius verschiebt.

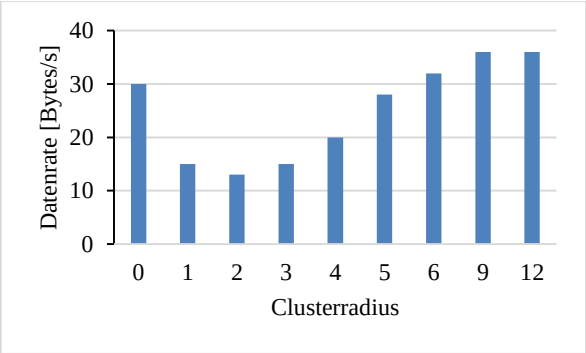

Abbildung 6.45: Minimale Signalisierungskosten mit Hierarchietiefe 3 (Ring)

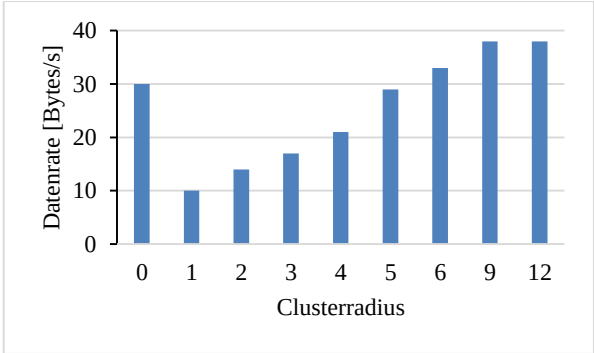

Abbildung 6.46: Minimale Signalisierungskosten mit Hierarchietiefe 4 (Ring)

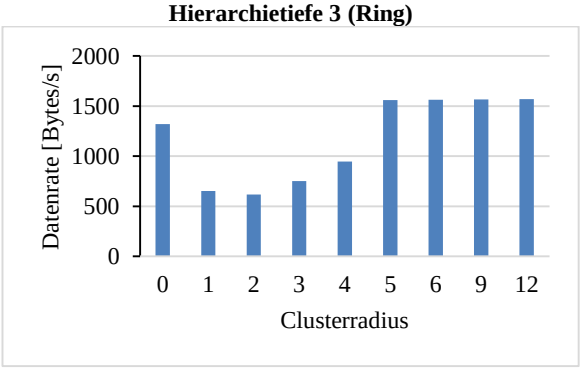

Abbildung 6.47: Maximale Signalisierungskosten mit Hierarchietiefe 3 (Ring)

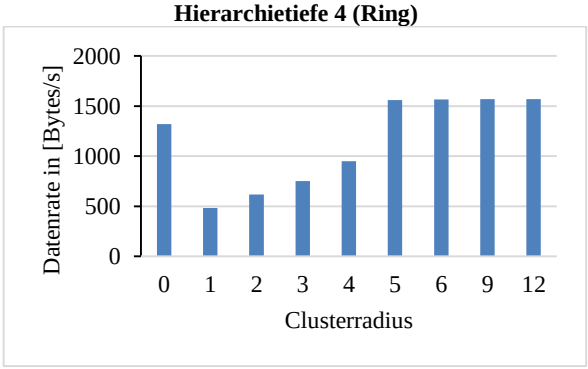

Abbildung 6.48: Maximale Signalisierungskosten mit Hierarchietiefe 4 (Ring)

In Abbildung 6.45 bis Abbildung 6.48 ist ein Vergleich zwischen Hierarchietiefe 3 und 4 zu sehen. Dabei ist zu erkennen, dass durch die ähnliche Struktur der erstellten Kontrollebene die Kurvenverläufe sehr ähnlich sind[20]. Dennoch fallen kleinere Unterschiede auf, sodass die minimalen Signalisierungskosten bei größerer Tiefe leicht erhöht ausfallen. Die Ursache sind die zusätzlich notwendigen *AnnounceCoordinator*-Nachrichten[21]. Die Ausnahme bildet dabei ein Clusterradius von 0, welcher für beide Hierarchietiefen aufgrund der sich daraus ergebenden Strukturen der Kontrollebene die gleiche Menge von Signalisierung zur Folge hat.

Bei Betrachtung der maximal zu erwartenden Kosten ist zu erkennen:

- Der Einfluss der Signalisierungen zur Koordinatorbekanntgabe ist im Vergleich zu den *RouteReport/RouteShare*-Nachrichten in diesem Fall sehr gering, sodass sich die ergebende Erhöhung der Kosten nicht erkennen lässt.
- Der prinzipielle Kurvenverlauf unterscheidet sich für einen Clusterradius von 1, sodass dieser Wert für eine Hierarchietiefe von 4 das neue Optimum für möglichst minimale Signalisierungskosten darstellt. Die Ursache liegt dabei in der veränderten Struktur der Kontrollebene und der Menge der darüber ausgetauschten Routingdaten.

Ergänzend zu diesen Untersuchungen der Hierarchietiefe ist in Anhang C eine Untersuchung für ein Netzwerk mit vierfacher Größe enthalten. Aus der Summe dieser Untersuchungen lässt sich schlussfolgern, dass die für möglichst kleine Signalisierungskosten optimalen Werte für Clusterradius und Hierarchietiefe stark von der Topologie (insbesondere dem Durchmesser) des Netzwerks abhängig sind.

[20] Im Vergleich dazu betragen die minimalen und maximalen Signalisierungskosten konstant 34 Bytes/s bzw. 1566 Bytes/s für eine Hierarchietiefe von 2. Die Ursache dafür liegt in der angewandten Verteilung von Koordinatorinstanzen der Kontrollebene, welche unabhängig des gewählten Clusterradius immer gleich ausfällt.

[21] Dies wurde durch zusätzliche Vergleichsmessungen bestätigt, welche an dieser Stelle zur besseren Übersichtlichkeit nicht aufgeführt sind.

Zusätzlich wird deutlich, dass eine Erhöhung der Hierarchietiefe keine sprunghafte Erhöhung der Maximalkosten verursacht.

6.3.1.6 Einfluss des Intervalls von *RouteReport*/*RouteShare*-Nachrichten

Als zweiter Einflussfaktor auf die Kosten der Kontrollebene ist die Entscheidung für die Intervalle einzelner Signalisierungen zu nennen. Variiert man diese für die Verteilung von Routingdaten von 1 Sekunde auf 5 Sekunden, ist eine lineare Abnahme der Kosten zu vermuten.

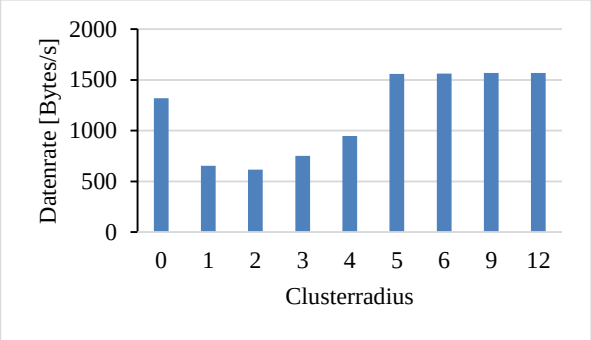

Abbildung 6.49: Maximale Signalisierungskosten mit einem Intervall von 1 Sekunde (Ring)

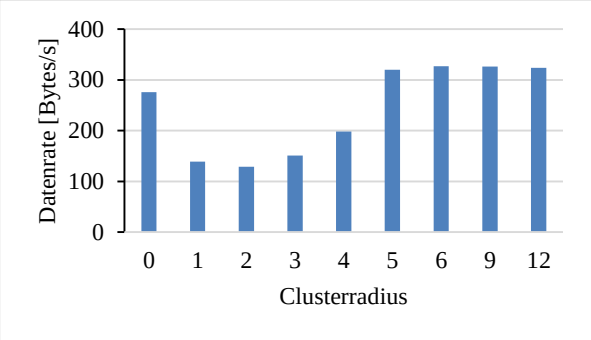

Abbildung 6.50: Maximale Signalisierungskosten mit einem Intervall von 5 Sekunden (Ring)

In Abbildung 6.49 ist nochmals das Ergebnis des letzten Abschnitts zu sehen, daneben ist in Abbildung 6.50 die Verteilung der verursachten Kosten für ein Intervall von 5 Sekunden zu sehen. Beide Darstellungen beziehen sich auf den Fall des maximalen Signalisierungsaufkommens. Berechnet man die jeweils durchschnittlich verursachte Datenrate, erhält man ein Verhältnis von etwa 4,8:1. Die Abweichung von dem erwarteten Verhältnis von 5:1 wird dabei durch den Einfluss von *AnnounceCoordinator*-Nachrichten verursacht, sodass die aufgestellte Vermutung einer linearen Abnahme bestätigt ist. Analog dazu kann der gleiche Zusammenhang ebenfalls für die Wahl des Intervalls von *AnnounceCoordinator*-Nachrichten durch Messungen gezeigt werden.

6.3.2 Speicheraufwand

Nachdem zuvor die Belastung der Links des Netzwerks untersucht wurde, behandelt dieser Abschnitt die durchschnittliche Belastung von Knoten bei Verwendung von HRM. Diese wird einerseits durch die für die Signalisierungen notwendigen Verbindungen verursacht, andererseits spielt die Größe des Routinggraphen eine maßgebende Rolle für die Belastung eines Knotens.

6.3.2.1 Einfluss des Clusterradius am Beispiel der Ringtopologie

Analog zur Untersuchung des Signalisierungsaufwands wird auch an dieser Stelle die Ringtopologie als typische Topologie eines Core-Netzwerks für die Untersuchung des Clusterradius verwendet. Wird ein Clusterradius von 0 oder ein sehr hoher Wert verwendet, ist mindestens ein Cluster mit sehr vielen Mitgliedern zu erwarten. Auf dem Knoten, auf dem die jeweils zugehörige Koordinatorinstanz des betroffenen Clusters instanziiert wurde, ist eine erhöhte Belastung zu erwarten.

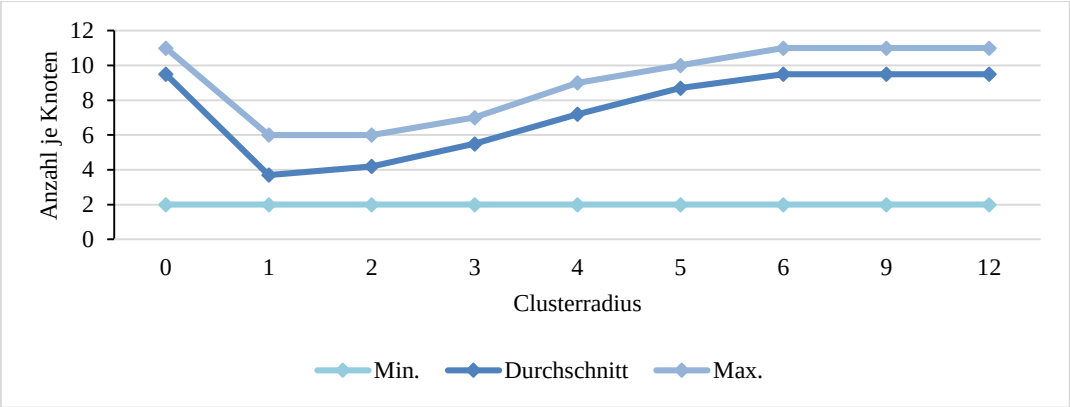

Abbildung 6.51: Anzahl von notwendigen Verbindungen der Kontrollebene (Ring mit 12 Knoten)

Zur genaueren Betrachtung der durchschnittlichen Belastung von Knoten zeigt Abbildung 6.51 die Anzahl von notwendigen Verbindungen. Da jeder Knoten stets zwei Nachbarknoten besitzt, mit denen er über die L0-Clustermanager Signalisierungen austauscht, wird dadurch ein Minimum von 2 verursacht. Das Maximum von notwendigen Verbindungen ist dagegen abhängig von der verwendeten Struktur der Kontrollebene – der höchste auftretende Wert ist 11. In diesem Fall hält der betroffene Knoten zu allen anderen eine separate Verbindung aufrecht. Ein Clusterradius von 1 oder 2 erscheint bei Auswertung von Abbildung 6.51 als sehr gute Wahl, wobei der optimale Wert bei 1 liegt.

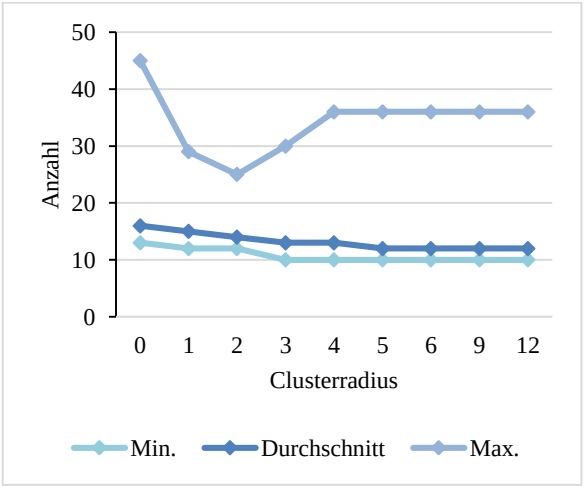

Abbildung 6.52: Knoten in den Routinggraphen (Ring mit 12 Knoten)

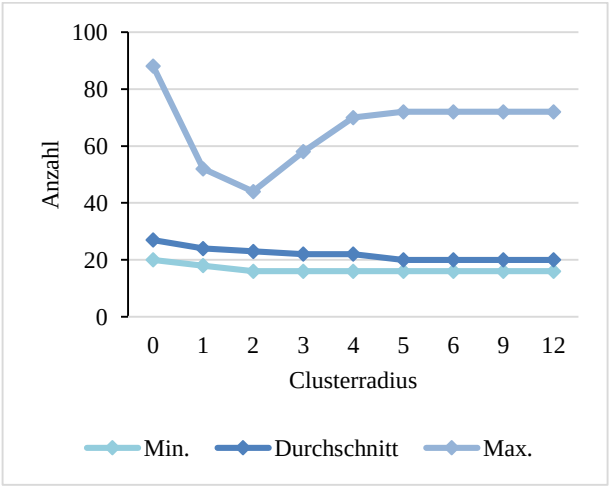

Abbildung 6.53: Kanten in den Routinggraphen (Ring mit 12 Knoten)

Abbildung 6.52 und Abbildung 6.53 zeigen den Einfluss des Clusterradius auf die resultierende Größe der Routinggraphen. Dabei ist zu erkennen, dass ein Radius von 2 den optimalsten Wert zur Begrenzung der auftretenden Größen für Routinggraphen darstellt. Dabei wird der größte Routinggraph typischerweise auf dem Knoten erstellt, der die Instanz des TOP-Koordinators beinhaltet.

Die Messungen dieses Abschnittes zeigen, dass analog der vorhergehenden Untersuchungen ein Wert von 0 als wenig sinnvoll erscheint, da dadurch viele Koordinatorinstanzen auf Hierarchielevel 1 und ein sehr großer Cluster für den TOP-Koordinator ausgebildet wird. Sehr hohe Clusterradien führen zu einem großen Cluster und somit zu erhöhter Belastung eines einzelnen Knotens. Anhang C bestätigt dies zusätzlich für eine größere Netzwerktopologie.

6.3.2.2 Einfluss der Topologie und der Anzahl von Knoten

Nachfolgend werden alle ausgewählten Basistopologie näher betrachtet und der dabei verursachte Speicheraufwand anhand der notwendigen Verbindungen der Kontrollebene und der Größe von gespeicherten Routinggraphen für eine steigende Anzahl von Knoten bewertet.

Ringtopologie

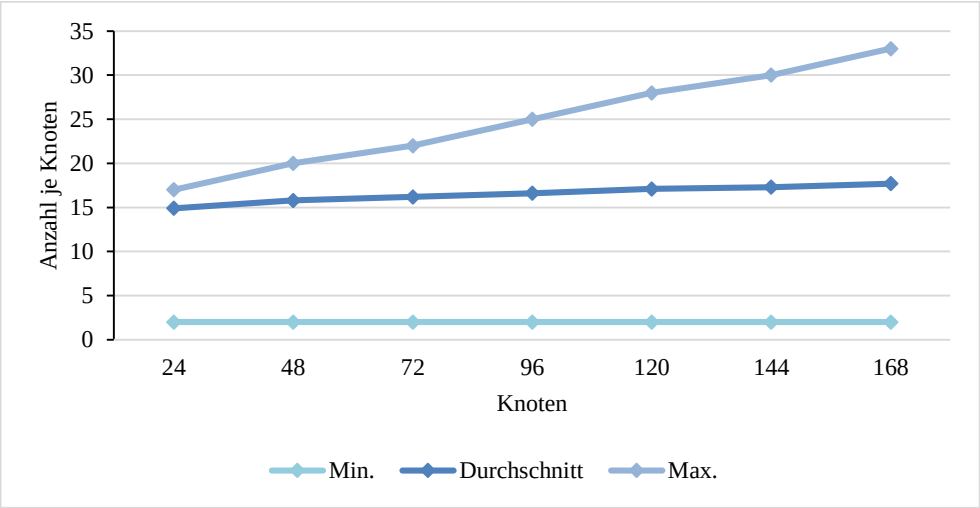

Abbildung 6.54: Anzahl von notwendigen Verbindungen der Kontrollebene (Ring)

Abbildung 6.54 zeigt die Zunahme von notwendigen Verbindungen bei einem Clusterradius von 8 für eine steigende Knotenanzahl. Daraus ist ein linearer Verlauf ersichtlich. Analog dazu steigen die durchschnittliche sowie die maximal auftretende Größe von Routinggraphen im Netzwerk ebenfalls linear an.

Maschentopologie

Die Anzahl der für jeden Knoten notwendigen Verbindungen der Kontrollebene steigt bei einer Maschentopologie linear mit steigender Knotenanzahl und kann explizit berechnet werden. Für *n* Knoten einer Masche ergeben sich jeweils *(n - 1)* Verbindungen für die Kommunikation der knotenlokalen Entitäten. Das Wachstum der HRG-Instanzen ist wiederum quadratisch aufgrund der hinzukommenden Anzahl von Links, welche für jeden weiteren Knoten zusätzlich im Graphen gespeichert werden müssen.

Sterntopologie

Die Anzahl der notwendigen Verbindungen für die Signalisierungen der Kontrollebene beschränkt sich bei den Gateway-Routern auf 1 und der zentrale Knoten hält als Gegenstück jeweils eine für jeden Gateway-Router aufrecht. Die Größen der lokalen HRG-Instanzen der Knoten wachsen wiederum linear mit zunehmender Anzahl von Gateway-Routern.

Broadcast-Domänen

Erst wenn einer der Knoten der Domäne mehr als eine Netzwerkschnittstelle besitzt, worüber weitere Netzwerke angeschlossen sind, wird ein Austausch von Routingdaten verwendet. In diesem Fall können die Endsysteme über Routen zu fremden Netzwerken mit Hilfe der Punkt-zu-Punkt-Verbindungen benachrichtigt werden[22]. Daraus ergibt sich für einen einzelnen Knoten eine lineare Verbindungkomplexität *O(n)* bezüglich der *n* Knoten einer angeschlossenen Broadcast-Domäne.

[22] Die Ausführungen des Abschnittes beziehen sich auf die aktuelle Implementierung. Während diese eher für Router mit Einzellinks zu direkten Nachbarroutern ausgelegt ist und separate Knoten-zu-Knoten-Signalisierungen verwendet, können alternativ auch Broadcast-Nachrichten verwendet werden. In dem Fall verhält sich die Datenrate innerhalb einer Domäne ausschließlich proportional zur Anzahl von bekannten Routen, explizite Verbindungen sind unnötig. Dadurch fällt jedoch die durch die vorher verwendeten Verbindungen bereitgestellte zuverlässige Übertragung von Signalisierungen weg. Dies bedarf weiterer Untersuchungen in zukünftigen Arbeiten.

Das Wachstum der HRG-Instanzen ist von den Eigenschaften der gewählten Implementierung abhängig und steigt für die aktuell verwendete Umsetzung von HRM für eine zunehmende Knotenanzahl innerhalb einer abgeschlossenen Domäne linear an.

6.3.3 Zusammenfassung

Die Ergebnisse der durchgeführten Untersuchungen bestätigen die korrekte Arbeitsweise der Signalisierungen und zeigen zugleich, dass der auftretende Kostenverlauf den Erwartungen entspricht:

- Die *RouteReport/RouteShare*-Nachrichten zur Verteilung von Routingdaten üben bei sehr hoher Netzdynamik den größten Einfluss auf die Gesamtkosten aus. Durch Senkung der verwendeten Signalisierungsrate können die dabei verursachten Gesamtkosten signifikant reduziert werden. Im Rahmen der Experimente dieser Arbeit wurden eher kurze Intervalle mit hohen Senderaten verwendet, um den *worst-case*-Fall näher zu untersuchen. In realen Anwendungen von HRM kann die Senderate dagegen auch geringer ausfallen. Dabei muss jedoch die Verzögerung bei der Aktualisierung von Routingdaten beachtet werden.
- Die Menge der *AnnounceCoordinator*-Nachrichten ist maßgebend für die minimal auftretenden Signalisierungskosten, welche bei konstanten QoS-Eigenschaften aller Routen verursacht werden.
- Der optimale Wert für den Clusterradius zur Reduktion der Kosten befindet sich zwischen 0 und dem Durchmesser des jeweiligen Netzwerks, ein Wert von 0 ist nicht sinnvoll. Die Anpassung des gewählten Wertes in Abhängigkeit vom Durchmesser des eingesetzten Netzwerks ist sinnvoll (aber nicht zwingend erforderlich für HRM).
- Aufgrund der bei HRM angewandten Netzwerkunterteilung und Topologieaggregation ergeben sich zusätzliche Vorteile für den durch HRM verursachten Signalisierungs- und Speicheraufwand.
- Durch weitere Hierarchielevels ist eine zusätzliche Kostenreduktion möglich.

Kriterium	**Ring**	**Masche**	**Stern**	**Domäne**
Signalisierungsaufwand (AnnounceCoordinator & RouteReport/RouteShare)	O(n)	$O(n^2)$	0	0
Speicheraufwand (Verbindungen)	O(n)	O(n)	• zentraler Router: O(n) • Gateway-Router: O(1)	O(n)
Speicheraufwand (HRG-Instanzen)	O(n)	$O(n^2)$	O(n)	O(n)

Tabelle 6.3: Signalisierungs- und Speicheraufwand in Abhängigkeit von der Knotenanzahl

Tabelle 6.3 zeigt eine Zusammenfassung über das allgemeine Kostenverhalten in Abhängigkeit von der Knotenanzahl *n*. Die durchschnittliche Linkbelastung gibt dabei Auskunft über die Skalierbarkeit des Signalisierungsaufkommens im Netzwerk und ist abhängig von der Anzahl der vorhandenen Routingziele, welche proportional zur Vergrößerung des Netzwerks zunimmt. Dies beeinflusst ebenfalls die resultierende Größe der HRG-Instanzen auf den Knoten des Netzwerks. Des Weiteren spielt die Art der Verlinkung eine wichtige Rolle, welche insbesondere im Fall der Maschentopologie für ein erhöhtes Nachrichtenaufkommen und eine rasche Vergrößerung von HRG-Instanzen sorgt. Die angegebene durchschnittliche Anzahl von Verbindungen zeigt die Skalierbarkeit des Managements der Kontrollebene. Sie entspricht dem anfangs vermuteten Verhalten einer linearen Zunahme in Abhängigkeit von der Knotenanzahl. Dabei wird jedoch durch die Verwendung des DCE-Algorithmus der Anstieg der Werte in Abhängigkeit vom gewählten Clusterradius signifikant gebremst.

6.4 Nutzbarkeit von Netzwerkressourcen auf Basis der Datenebene

Zur Bewertung des Nutzens von HRM ist insbesondere ein Vergleich zu reinem BE-basiertem Routing sinnvoll. Dabei stehen folgende Vorteile von HRM im Fokus:

- **Beachtung von Qualitätsanforderungen:** Jede Anwendung kann Anforderungen an die Übertragung ihrer Daten definieren und an das Routing signalisieren.
- **Beachtung von aktuellen QoS-spezifischen Routeneigenschaften:** Die Routingentscheidungen des durch die Datenebene bereitgestellten dynamischen Routings sind immer abhängig von der aktuell bekannten Kapazitätsverteilung im Netzwerk, sodass Veränderungen in den verfügbaren Linkressourcen über die ausgetauschten Routingdaten und die Routingtabellen im Netzwerk nachfolgende Routenberechnungen beeinflussen.
- **Fairness gegenüber möglichen parallelen Übertragungen:** Primär wird nach der WSPF-Strategie die Route zu einem Ziel ausgewählt. Wenn die gefundene Lösung nicht den Qualitätsanforderungen der Anwendung genügt, wird die SWPF-Strategie angewandt und die Daten entlang der Route mit der größten noch verfügbaren Datenrate geleitet.

Durch Versuche sollte der tatsächlich durch HRM erreichte Gewinn bei der Nutzung von Netzwerkressourcen ersichtlich werden. Dabei ist auch die Frage interessant, inwiefern die Aggregationsmechanismen beim Austausch von Route*Report/RouteShare*-Nachrichten zu Einschränkungen im Routing und damit wiederum zu Beeinträchtigungen bei der Nutzung von Netzwerkressourcen führen.

Die nachfolgenden Betrachtungen konzentrieren sich auf das IntServ-Modell und dem Routing von initialen Verbindungsanfragen. Folgende Annahmen sind dabei enthalten:

- **Festes Routing:** Alle versendeten Pakete eines Datenstroms verwenden die bei der Weiterleitung der initialen Verbindungsanfrage einmalig festgelegte Route.
- **Strombasierte Ressourcenverwaltung:** Die im Netzwerk als „verfügbar“ geltenden Linkkapazitäten verringern sich mit zunehmenden Datenströmen infolge neuer Reservierungen[23].
- **Unmittelbare Aktualisierung von Routingdaten:** Die in realen Netzwerken auftretenden Verzögerungen bei der Aktualisierung von Routingdaten werden vernachlässigt, sodass HRM unter optimalen Bedingungen betrachtet wird. Durch diese Vereinfachung werden nachvollziehbare Ergebnisse erzielt, die nicht durch sporadische Verzögerungen eher „zufällig“ erscheinen.
- ***Best Effort*-Pakete und Pakete mit *DiffServ*-basierten Anforderungen haben keinen Einfluss auf die Ressourcenverwaltung:** Pakete ohne eine feste Reservierung werden nicht näher betrachtet und beeinflussen den Inhalt von Routingtabellen nicht. Ihre Weiterleitung ist implementierungsspezifisch: Sie können entweder mit Hilfe von BE-Routing oder dem HRM-spezifischen Routing durch das Netzwerk geleitet werden. Letzteres bietet sich für Pakete mit *DiffServ*-basierten Qualitätsanforderungen (die Anforderungen sind bspw. durch die DiffServ-Klasse gegeben) an, um ungeeignete Routen im Netzwerk zu vermeiden und dadurch Paketverluste (infolge von Überlastsituationen) vorbeugend zu verhindern.

Zur besseren Übersichtlichkeit konzentrieren sich die nachfolgenden Messungen auf die Ringtopologie und alle weiteren Basistopologien werden diskutiert. Die Ringtopologie stellt dabei die kostengünstigste

[23] Diese werden durch einen Reservierungsmechanismus zum Start einer Verbindung ausgelöst. Während man für ein IP-basiertes Netzwerk typischerweise RSVP einsetzt, kommen innerhalb der verwendeten FoGSiEm-Implementierung die FoG-spezifischen Mechanismen zum Einsatz. Sie üben jedoch keinen Einfluss auf die erzielten Ergebnisse aus. Bei jeder Reservierung wird die Kontrollebene automatisch informiert, sodass Signalisierungen ausgelöst werden und dadurch automatisch die Routingdaten sowie die Routingtabellen im Netzwerk aktualisiert werden.

Variante zur Verwendung von redundanten Routen dar und ist dadurch sehr häufig in Core-Netzwerken zu finden.

6.4.1 Einfluss des Clusterradius am Beispiel der Ringtopologie

Zur Analyse des Einflusses des Clusterradius wird im Folgenden erneut eine Ringtopologie mit 12 Knoten verwendet, die physikalisch maximal verfügbare Datenrate jedes Links wird dabei mit 100 Mbit/s angenommen. Die vorgestellten Werte stellen den jeweiligen Mittelwert aus 100 Versuchen dar. Für jeden Versuch wurde folgender Ablauf angewandt:

1.) **Initialisierung:** Das Netzwerk wird mit 50 Verbindungen zwischen jeweils zufällig[24] ausgewählten Knotenpaaren initialisiert. Jede dieser Verbindungen reserviert entlang der jeweiligen Route eine individuelle Datenrate zwischen 0 und 1 Mbit/s.
2.) **Nutzverbindungen:** Im nächsten Schritt werden zufällig zwei[25] Knoten Q und Z ausgewählt, zwischen denen kontinuierlich neue Verbindungen mit einer geforderten Datenrate von 1 Mbit/s unter Verwendung des HRM-basierten Routings aufgebaut werden. Dies wird 200-fach ausgeführt, sodass dadurch für das Szenario die maximal mögliche Anzahl von erfolgreichen Verbindungen mit erfüllten Qualitätsanforderungen erfasst wird.
3.) **Statistik:** Die Anzahl von erfolgreich gestarteten Verbindungsversuchen (bei denen die Qualitätsanforderungen erfüllt sind) werden ermittelt und in der globalen Statistik für HRM-Routing gespeichert.
4.) **Wiederholung:** Alle Verbindungen (exklusive der initialen Verbindungen) werden geschlossen und reservierte Ressourcen im Netzwerk wieder freigegeben. Danach werden die Verbindungen aus Schritt 2 erneut zwischen Q und Z aufgebaut und nachfolgend die erfolgreich gestarteten gezählt (analog zu Schritt 3). Statt HRM-Routing wird bei diesem Durchlauf BE-Routing verwendet und die globale Statistik am Ende entsprechend aktualisiert.

Aus den gewonnen Daten lässt sich das Verhältnis zwischen dem Nutzen von HRM- und BE-basiertem Routing ermitteln. Es ist anzunehmen, dass HRM maximal die doppelte Anzahl von Verbindungen (theoretisches Optimum) zwischen den Knoten Q und Z ermöglicht, da es im Gegensatz zu BE-Routing beide Pfade innerhalb der Ringtopologie für das Routing von Anwendungsdaten ausnutzt.

[24] Für die Generierung von zufälligen Auswahlen oder Zahlwerten verwendet die Implementierung die Java-Klasse „Random“, deren Ergebnisse der stetigen Gleichverteilung (auch „Uniformverteilung“) unterliegen und Werte für ein vorgegebenes Intervall liefern.

[25] An dieser Stelle wäre es auch denkbar, mehre verschiedene Knotenpaare zu wählen und die jeweils genutzten Netzwerkressourcen (erfolgreiche Verbindungen) zu betrachten. In diesem Fall wäre der Zugewinn von HRM gegenüber BE-Routing – neben der Topologie – zusätzlich von den jeweils ausgewählten Knotenpaaren und der für jede Verbindung angefragten Datenrate abhängig, sodass es günstige und ungünstige Konstellationen mit jeweils individueller Blockierungswahrscheinlichkeit für nachfolgende Verbindungsanfragen geben kann. Dies würde den Vergleich der erfolgreichen Verbindungen im Gegensatz zum verwendeten eher einfachen Versuchssetup erschweren.

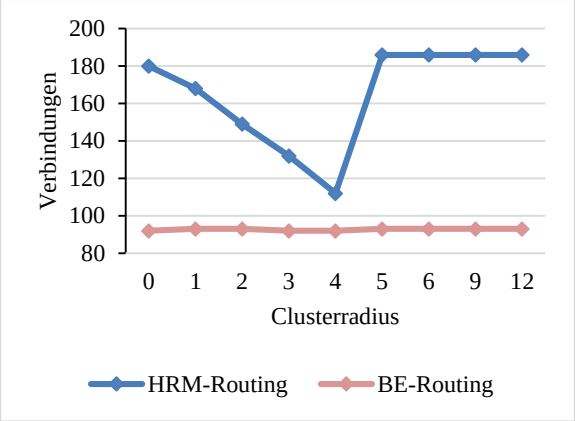

Abbildung 6.55: Erfolgreiche Verbindungen

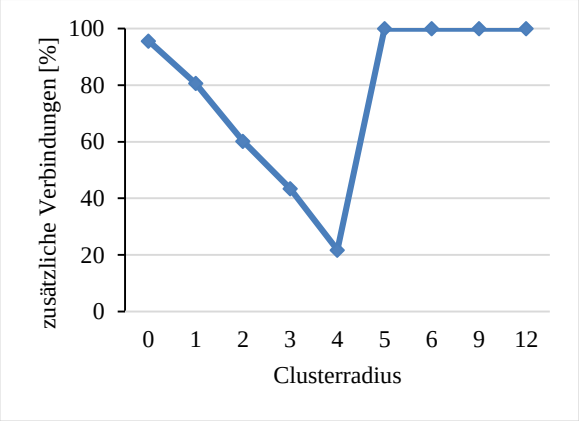

Abbildung 6.56: Ressourcengewinn durch HRM-Routing

Abbildung 6.55 zeigt den Einfluss des Clusterradius und stellt jeweils die Anzahl der erfolgreichen Verbindungen unter Beachtung der Qualitätsanforderungen für HRM- und BE-basiertes Routing dar. Die Werte entsprechen der akkumulierten maximalen Datenrate, welche maximal (mit Hilfe von zwei aufeinanderfolgenden Reservierungen) im Netzwerk zwischen Q und Z zur Verfügung stehen würde. Ausgehend von einem Clusterradius 0 bis zu einem Wert von 4 verringert sich dieser Maximalwert stetig, um ab einen Wert von 5 konstant bei 186 erfolgreiche Reservierungen mit jeweils 1 Mbit/s (entspricht 186 Mbit/s maximaler Datenrate) zu verbleiben. Die Ursache dieses Verhaltens liegt in der resultierenden Größe der jeweils existierenden L1-Cluster und der innerhalb der Implementierung integrierten Vermeidung von Routingschleifen. Dadurch werden jedem Knoten Q keine Routen zum Ziel Z signalisiert, welche fremde Cluster[26] durchqueren. Für einen Clusterradius von 4 tritt dieser Fall besonders häufig ein, da die ausgebildeten L1-Cluster dabei einen besonders großen Durchmesser besitzen und somit die ausgewählten Knoten (Q und Z) mit hoher Wahrscheinlichkeit dem gleichen L1-Cluster angehören. Das Routing der Datenebene hält dabei die Datenübertragungen innerhalb des jeweiligen L1-Clusters, obwohl alternative Wege durch einen anderen L1-Cluster existieren. Wie aus Abbildung 6.56 ersichtlich, verringert sich dadurch der durch HRM erhaltene Ressourcengewinn. Wählt man statt der bisherigen Werte einen Clusterradius größer 4, enthält die Kontrollebene in diesem Szenario ausschließlich einen L1-Koordinator, der die Routingdaten zwischen seinen untergeordneten L0-Koordinatoren verteilt. Dadurch nutzt das HRM-Routing beide jeweils zwischen Q und Z existierenden Routen, sodass gegenüber BE-Routing ein Gewinn von etwa 100% bei den Messungen erzielt wird.

Obwohl das vorgestellte Verhalten scheinbar Ressourcen ungenutzt lässt, ist es für die Umsetzung von Routingpolitiken notwendig und wird im nachfolgenden Abschnitt 6.4.4 verwendet, um das Netzwerk in sogenannte Routingzonen aufzuteilen.

6.4.2 Einfluss von Routingschleifen auf höheren Hierarchielevels

Werden Routingschleifen durch fremde Cluster erlaubt, wird dadurch die Nutzung aller Ressourcen des Netzwerks für jeden Knoten unabhängig des gewählten Clusterradius unterstützt. Angewandt auf die Ringtopologie mit 12 Knoten führt dies entsprechend Abschnitt 3.8.2.2 zu folgendem Verhalten:

- Die kürzeste Route zum Ziel wird verwendet, solange sie
  a. die Qualitätsanforderungen der Anwendung erfüllt,
  b. ihre Auslastung einen definierten Wert $util_{max}$ nicht überschreitet und
  c. die weiterhin verfügbare Datenrate einen definierten Wert dr_{min} nicht unterschreitet.

[26] Wenn Q und Z in einem gemeinsamen Cluster liegen, dann sind alle anderen Cluster des gleichen Hierarchielevels als „fremde Cluster“ zu verstehen.

- Andernfalls wird die Route mit den besten Qualitätseigenschaften gewählt. Dabei wird die in Abschnitt 3.8.3 erläuterte Prioritätsverteilung zwischen den einzelnen Kriterien angewandt.

Folglich werden die neuen Verbindungen zuerst ausschließlich entlang des kürzesten Pfades geleitet, um anschließend den Pfad mit der größten verfügbaren Kapazität zu nutzen, sodass mit zunehmender Anzahl von Verbindungen alle verbleibenden Ressourcen im Netzwerk ausgenutzt werden.

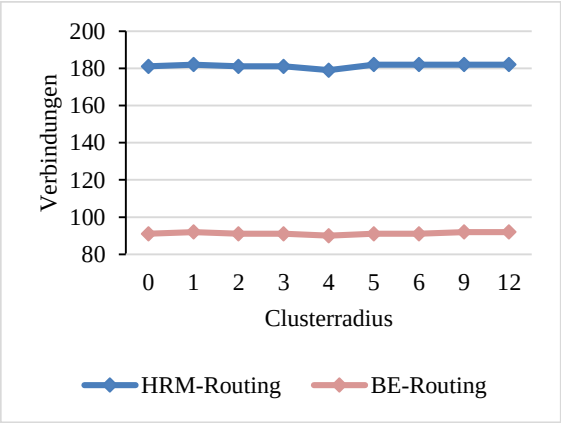

Abbildung 6.57: Erfolgreiche Verbindungen (mit Routingschleifen)

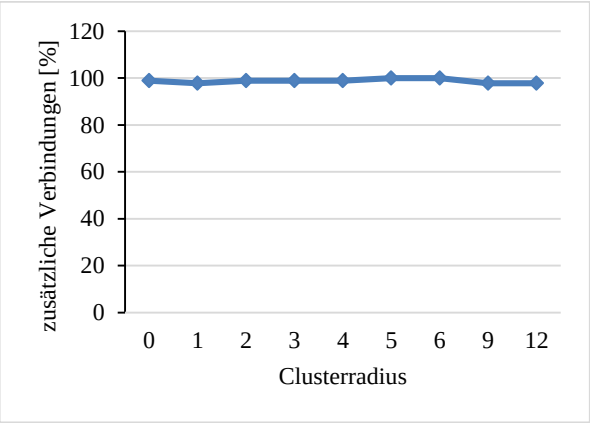

Abbildung 6.58: Ressourcengewinn durch HRM-Routing (mit Routingschleifen)

Die Verwendung von Routingschleifen wurde durch Versuche untersucht. Abbildung 6.57 zeigt die dabei erhaltenen erfolgreichen Verbindungen in Abhängigkeit vom jeweils verwendeten Routing. Beim Vergleich zwischen beiden Varianten wird deutlich, dass HRM im Vergleich zu BE-Routing jeweils beide Pfade zwischen Q und Z verwendet. In Abbildung 6.58 ist zusätzlich der durch HRM resultierende Ressourcengewinn gegenüber BE-Routing dargestellt. Im Gegensatz zu Abschnitt 6.4.1 ist er unabhängig des verwendeten Clusterradius mit etwa 100% für das Szenario zu erkennen.

6.4.3 Einfluss der Topologie

Bei der Ring-, Maschen- und Sterntopologie existieren redundante Pfade zu Zielen und die Vorteile von HRM können für die Übertragung von Anwendungsdaten ausgenutzt werden. Der dabei erreichte Gewinn gegenüber BE-Routing ist abhängig von der Menge von existierenden alternativen Pfaden zum jeweiligen Ziel. Bei der Maschen- und Sterntopologie variiert diese Anzahl und die Struktur der Alternativrouten in Abhängigkeit zum gewählten Knotenpaar (Quelle und Ziel), sodass einzig die Ringtopologie unabhängig des gewählten Knotenpaars immer zwei nutzbare Pfade zu einem Ziel bietet. Diese sind sogar immer vollständig disjunkt zueinander und eigenen sich besonders für ein *Multipath Routing*.

Broadcast-Domänen unterscheiden sich grundsätzlich von den anderen Basistopologien. Sie können zwar auf Basis von Switches redundante Pfade beinhalten, diese sind jedoch für die Topologieerkennung der Kontrollebene nicht ersichtlich. Unabhängig des Einsatzes von HRM wird in heutigen Netzwerken diese Form der Schleifenbildung sehr häufig durch den Einsatz des *Spanning Tree Protocols* (STP) [148] oder seines Nachfolgers namens *Shortest Path Bridging* (SPB) [149] verhindert, sodass Pakete generell nur entlang einem der vorhandenen Pfade zum Ziel geleitet werden können.

6.4.4 Routingzonen zur Umsetzung von Netzwerkrichtlinien

In Abhängigkeit von den Zielen des Netzwerkbetreibers ist es denkbar, dass ein vorhandenes Netzwerk in sogenannte Routingzonen unterteilt werden soll, wobei strikt zwischen internem und externem Datenverkehr unterschieden werden soll. Ähnlich BGP und OSPF muss dafür auch bei HRM ein manueller Eingriff in die Konfiguration erfolgen, um ein solches Szenario umzusetzen.

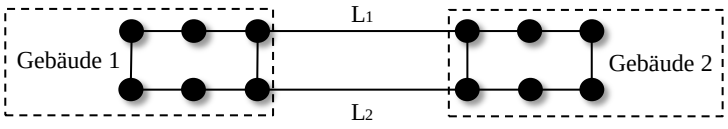

Abbildung 6.59: Anbindung von zwei Gebäuden bei gleichzeitiger Unterteilung in Routingzonen

Abbildung 6.59 zeigt ein Beispielszenario, in dem jedes Gebäude eine Ringtopologie besitzt und beide Gebäude über 2 Links miteinander verbunden sind. Diese Anbindung kann beispielsweise durch den Einsatz von Richtfunk realisiert sein, wobei die verfügbare Datenrate entlang solcher Links im Gegensatz zur kabelbasierten Links typischerweise eher geringer ist. Dadurch ist es sinnvoll, möglichst wenig gebäudeinternen Datenverkehr darüber zu übertragen. Durch den Einsatz der in Abschnitt 6.4.1 vorgestellten Variante des Routings kann diese Einschränkung explizit umgesetzt werden. Zusätzlich ist es notwendig, die Netzwerkunterteilung, welche im Fall von HRM der Clusterbildung entspricht, auf höheren Hierarchielevels zu konfigurieren. Dafür genügt es, die Ausbreitung der *AnnounceCoordinator*-Signalisierungen auf jeweils einem Knoten der beiden Links L_1 und L_2 zu begrenzen, sodass die Bekanntgabe von L0-Koordinatoren auf jeweils eines der beiden Gebäude beschränkt ist, die Bekanntgabe der pro Gebäude platzierten L1-Koordinatoren darf dabei nicht beeinflusst werden[27]. Die Kontrollebene beinhaltet somit für jedes Gebäude einen separaten L1-Cluster, deren Koordinatoren wiederum einem global eindeutigen TOP-Koordinator untergeordnet sind. Als Resultat erhalten Knoten von Gebäude 1 keine Kenntnis über Routen, welche durch Gebäude 2 wiederum zu Gebäude 1 führen. Zusätzlich werden keinem Knoten interne Netzwerkdetails des jeweils anderen Gebäudes signalisiert.

Dieses Szenario konnte mit Hilfe der Implementierung nachgestellt werden. Dabei wurden mit Hilfe der in Anhang D.3 abgebildeten grafischen Oberfläche entsprechende Einstellungen auf 2 Knoten vorgenommen, sodass daraufhin die *AnnounceCoordinator*-Nachrichten in ihrer Ausbreitung eingeschränkt wurden und die Kontrollebene sich wie gewünscht automatisch umstrukturierte.

6.4.5 Diskussion von Auswirkungen der Topologieaggregation

Da bei HRM für die Signalisierung von Routingdaten aggregierte Beschreibungen der vorhandenen Topologie verwendet werden, kann dies zu Auswirkungen im Routing führen.

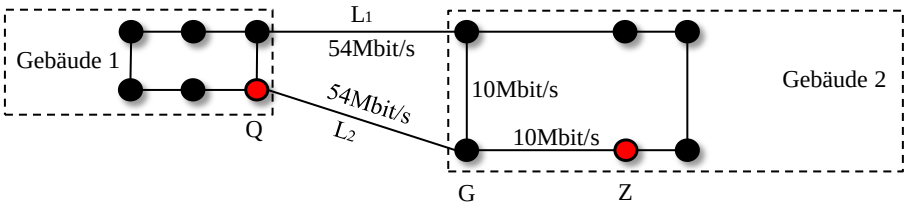

Abbildung 6.60: Auswirkung von Topologieaggregationen (QoS-Eigenschaften)

Abbildung 6.60 zeigt ein Beispielszenario, bei dem Daten von Q nach Z mit einer Datenrate von 20 Mbit/s übertragen werden sollen. Die meisten Links besitzen dabei eine maximale Datenrate von 100Mbit/s, wobei einige eine geringere Kapazität von 10 oder 54 Mbit/s aufweisen. Analog zu Abschnitt 6.4.4 sind Routingschleifen verboten und die Ausbreitung von *AnnounceCoordinator*-Nachrichten ist explizit begrenzt, sodass jedes Gebäude als eigenständiges Netzwerk mit einem jeweiligen L1-Koordinator gilt. Die durch HRM bestimmte Routingentscheidung auf Q würde in diesem Fall den Link L_2 bevorzugen, da die lokale Routingtabelle die Route über L_2 mit maximal 54 Mbit/s beinhaltet. Angekommen auf Knoten G wäre eine sinnvolle Fortsetzung der Reservierung für die anstehende Verbindung nicht möglich. Im Gegensatz dazu wäre ein Routing über den Link L_1 erfolgreich. Diese Route gilt

[27] Siehe Abschnitt 3.3.5.2

jedoch aufgrund des zusätzlichen Zwischenknotens bis zum Erreichen von Gebäude 2 als teurer und der Engpass um Knoten G ist den Knoten von Gebäude 1 nicht bekannt. In diesem Fall schlägt das Routing der Datenebene fehl, obwohl die notwendigen Ressourcen im Netzwerk verfügbar sind. Zur Verbesserung des Routings für solche Szenarien bieten sich die in Abschnitt 2.3.2 beschriebenen *Selective Probes* an, wodurch Engpässe im Netzwerk ermittelt und durch Alternativrouten umgangen werden können. Ihr Einsatz ist jedoch Bestandteil zukünftiger Erweiterungen und nicht dieser Arbeit in detaillierter Form enthalten.

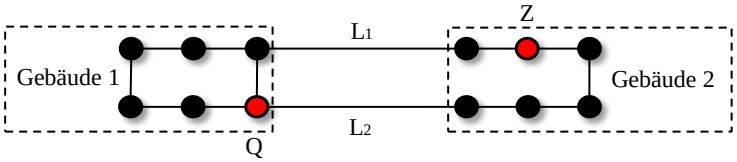

Abbildung 6.61: Auswirkung von Topologieaggregationen (Hop-Distanz)

Des Weiteren ist es möglich, dass aufgrund von Topologieaggregation die falsche Route als scheinbar kürzeste zu Knoten innerhalb eines fremden Clusters erscheint, obwohl durch die interne Struktur des Zielclusters die Wahl eines alternativen Pfads aus globaler Sicht zu einer kürzeren Route führt. Abbildung 6.61 zeigt das zuvor verwendete Szenario in modifizierter Form: Erneut wurde für jedes Gebäude eine separate Routingzone festgelegt. Die Datenebene von HRM wählt in diesem Fall auf dem Knoten Q jedoch fälschlich die Route über den Link L_2 als kürzeste zum Ziel Z aus, da die notwendigen Details über die Beschaffenheit des Netzwerks von Gebäude 2 fehlen. Die Auswahl des Pfades über den Link L_1 stellt das tatsächlich beste Ergebnis für eine möglichst kleine Hop-Distanz dar. Somit kann die bei WSPF-Routing ausgewählte Route gegenüber der tatsächlich kürzesten länger sein, der *stretch factor*[28] ist somit größer als 1. Er wird maßgeblich durch den Durchmesser d_{Ziel} des jeweiligen Zielclusters beeinflusst. Ein weiterer Einflussfaktor ist die Topologieaggregation während der Bestimmung von Routen durch einen Cluster[29]: Die Länge des bei Benutzung der Route tatsächlich verwendeten Pfads durch einen Cluster kann von der signalisierten Hop-Distanz abweichen.

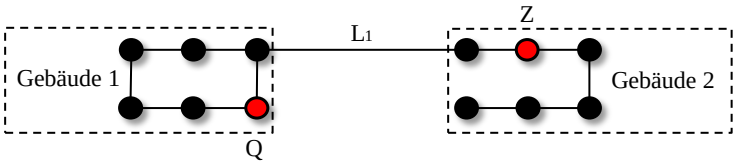

Abbildung 6.62: Auswirkung von Topologieaggregationen (Multipath)

Abbildung 6.62 zeigt das bisherige Szenario in nochmals modifizierter Form, sodass zwischen beiden Gebäuden nur noch ein Link existiert, es gibt erneut zwei Routingzonen. Durch die Signalisierungen von HRM enthält die Routingtabelle von Q zwei Einträge, welche jeweils zu Gebäude 2 und dem dortigen Zielknoten Z führen. Soll die maximale Datenrate zwischen beiden Knoten verwendet werden, ist für den Routingalgorithmus aus der lokalen Routingtabelle nicht ersichtlich, dass beide Pfade den Link L_1 verwenden. Werden beispielsweise Links mit einer maximalen Datenrate von 100 Mbit/s verwendet, ist bereits nach Reservierung der Ressourcen für die erste Verbindung mit 100 Mbit/s der Maximalwert erreicht. Aufgrund der Verzögerungen bei den Signalisierungen der Kontrollebene kann die zweite Route zwischen beiden Knoten jedoch weiterhin als vollständig verfügbar gelten und eine Anforderung einer zweiten Verbindung mit 100 Mbit/s wird fehlschlagen.

[28] Siehe Abschnitt 2.3.3
[29] Siehe Abschnitt 3.5.2.2

6.4.6 Zusammenfassung

Die vorgestellten Untersuchungen zur Evaluierung der Datenebene haben die Vorteile von HRM gegenüber reinem BE-basiertem Routing gezeigt. Mit Hilfe des neuen Routingmanagements ist es möglich, in Abhängigkeit von den gegebenen Qualitätsanforderungen mehrere verschiedene Pfade zwischen Quelle und Ziel zu nutzen, sodass die aktuell vorhandenen Kapazitäten mehrerer Routen gleichzeitig für eine Anwendung zur Verfügung stehen. HRM steht dabei als autonom arbeitendes Routingmanagement zur Verfügung, zugleich ist es jedoch auch an die Bedürfnisse des Netzwerkbetreibers anpassbar. Er kann mit Hilfe von Netzwerkrichtlinien festlegen, inwiefern einzelne Knoten alle Teile des Netzwerks für ihre Verbindungsanfragen verwenden dürfen. Dabei ist insbesondere die Verwendung von Routingschleifen für die Performanz der Datenebene von HRM entscheidend. Werden diese nicht verwendet, sind der verwendete Clusterradius und die daraus resultierende Größe von erstellten Clustern für die Performanz von HRM wichtige Einflussfaktoren. Je größer der Clusterradius gewählt wird, desto mehr Details werden abstrahiert und bei der Verteilung von Routingdaten nicht übertragen. Dadurch wird einerseits die Menge der signalisierten Daten reduziert, andererseits wird aber auch die Nutzbarkeit von Netzwerkressourcen beeinflusst. Werden fremde Cluster jedoch einbezogen, bietet das Routing der Datenebene von HRM im Vergleich zu BE-Routing den größten Vorteil, der unabhängig vom gewählten Clusterradius ist.

Fremde Cluster können ausdrücklich vom Netzwerkbetreiber unerwünscht und durch definierte Netzwerkrichtlinien verboten sein. Bei der Umsetzung von solchen Vorgaben spielt bei HRM insbesondere die Vermeidung von Routingschleifen und das Anwenden von Filterregeln für *AnnounceCoordinator*-Nachrichten eine Rolle. Abhängig von den Funktionen der Implementierung muss ein solches Setup entweder manuell durch den Netzwerkadministrator an den Schlüsselpunkten im Netzwerk konfiguriert oder über externe automatische Systeme vorgegeben werden. Dadurch lässt sich die Ausbildung und Größe von Clustern beeinflussen und somit auch die durch die Kontrollebene verteilten Informationen über vorhandene Routen. Jedoch müssen die während der Verteilung von Routingdaten angewandten Aggregationsmechanismen bei der Festlegung von Netzwerkrichtlinien beachtet werden. Sie haben Auswirkungen auf das resultierende Routingverhalten.

6.5 Diskussion zur Wahl der Hierarchietiefe und des Clusterradius

Wie innerhalb von Abschnitt 6.3 gezeigt und durch Anhang C ergänzt, kann sowohl durch Anpassung des Clusterradius als auch der Hierarchietiefe das aufkommende Signalisierungsaufkommen und die damit verbundenen Kosten von HRM positiv beeinflusst werden. Dabei sind folgende Regeln zur Optimierung zu beachten:

- mit steigendem Durchmesser des Netzwerks steigt der Wert für den optimalen Clusterradius
- mit steigender Hierarchietiefe verkleinert sich der Wert für den optimalen Clusterradius

Bei der Wahl der Hierarchietiefe ist zusätzlich zu beachten, dass jedes weitere Hierarchielevel zu weiteren Verzögerungen führen kann, sodass entsprechend Abschnitt 3.10.6 die maximal zu erwartende Verzögerung bis zur Aktualisierung lokaler Routingdaten steigt.

Für eine Hierarchietiefe von 2 degradiert die Signalisierungen der Kontrollebene zu einer einfachen Client-Server-Kommunikation, welche ähnlich zu OSPF und seinem Designated Router arbeitet, sodass jegliche Signalisierungen (unabhängig von der Anzahl der dafür zu passierenden Knoten) mit einem zentralen Server ausgetauscht werden. In dieser Arbeit wird folglich eine Hierarchietiefe größer 2 favorisiert, da dadurch sowohl eine gute Datenreduktion als auch Skalierbarkeit für größere Netzwerke möglich wird. Eine Hierarchietiefe von 3 verursacht dabei im Vergleich zu größeren Werten die geringste Verzögerung und stellt zugleich die kleinste Hierarchie dar. Somit wird dieser Wert in dieser Arbeit

empfohlen, die empirisch ermittelten Kostenverläufe lassen für diese Wahl keine signifikanten Nachteile während der Betriebsphase erkennen.

Für die Wahl des Clusterradius stellte sich für eine Ringtopologie mit einem Durchmesser von 6 ein Wert von 2 als Optimum für möglichst geringe Signalisierungskosten heraus. Dies stellt jedoch nur ein Beispiel dar und kann nicht verallgemeinert werden, die optimale Lösung ist dabei stark von der Topologie und der darin vorhandenen Konnektivität von einzelnen Knoten abhängig. Des Weiteren müssen die Erkenntnisse von Abschnitt 6.4.1 bei der Wahl des Clusterradius beachtet werden, andernfalls kann es zu Einschränkungen in der Nutzbarkeit von Netzwerkressourcen kommen.

6.6 Vollständigkeits- und Konsistenztest mit realen Paketen

Häufig verbleibt zum Ende der Betrachtung von Simulationsergebnissen die Frage, inwiefern die vorgestellten Signalisierungen tatsächlich für reale Umgebungen in der vorgestellten Form genügen. Bei der Beantwortung dieser Frage spielt sowohl die Vollständigkeit der eingesetzten Synchronisationsschritte als auch die Konsistenz der durch sie an beiden Enden der Kommunikation aktuell gehaltenen Statusdaten eine entscheidende Rolle.

Zur Überprüfung wurden zwei eigenständige Computer verwendet, auf jedem wurde ein Ringnetzwerk aus 6 Knoten simuliert, welches sinnbildlich für das Core-Netzwerk eines Gebäudes stand. Zusätzlich wurden beide Gebäude mit Hilfe der realen Netzwerkschnittstellen und der Emulatorfunktion von FoG-SiEm miteinander verknüpft, sodass darüber Daten ausgetauscht werden konnten. Aus Sicht des HRM-Routingdienstes erschien das resultierende Netzwerk als zusammenhängend und die Unterschiede zwischen Simulation und Emulation waren nicht ersichtlich.

Die resultierenden Statusdaten wurden mit Hilfe der in Anhang D abgebildeten grafischen Anzeigen überprüft und zeigten ein korrektes Ergebnis. Zusätzlich wurden kontinuierlich Testverbindungen mit Hilfe von explizit gestarteten Instanzen der *QoSTestApp* gestartet und die Ergebnisse der Routingentscheidungen mit wachsender Auslastung des Netzwerks überprüft. Dabei wurde erneut die korrekte Arbeitsweise der Implementierung bestätigt.

6.7 Schlussfolgerungen

In Kapitel 6 wurde das Verhalten von HRM anhand von ausgewählten Szenarien durch praktische Experimente und Messungen untersucht. Dabei wurde die in Kapitel 4 vorgestellte Implementierung der Kontroll- und Datenebene verwendet, welche die Konzeption von HRM vollständig umsetzt. Statt auf einzelnen komplexeren Topologien lag der Fokus bei den Untersuchungen auf ausgewählten Basistopologien, welche aus Sicht des Autors dieser Arbeit typische Grundbausteine von Netzwerktopologien darstellen. Auf Basis der dadurch gewonnenen Erkenntnisse sind Schlussfolgerungen für komplexere Netzwerke möglich, deren Struktur sich aus der Kombination von Basistopologien ergibt.

Die Evaluierung hat bestätigt, dass die Protokolle der Kontrollebene für alle ausgewählten Basisstrukturen korrekt funktionieren und die dabei erreichten Ergebnisse den Erwartungen entsprechen: Die Signalisierungen führen zur gewünschten Verteilung von Managementinstanzen, Adressen sowie Routingdaten. Auf dieser Basis werden Routingtabellen auf allen Knoten eines Netzwerks erstellt, sodass Pakete von jedem Knoten zu jedem anderen weitergeleitet werden können. Bei der Untersuchung der **Startphase** der Kontrollebene wurde festgestellt, dass die verursachten Signalisierungen mit zunehmender Knotenanzahl für die Basistopologien überwiegend linear ansteigen. Die Ausnahme bildet dabei die Maschentopologie, welche aufgrund der auftretenden quadratischen Zunahme von Links bzw. Pfaden zu einem quadratischen Anstieg der notwendigen Nachrichten führen.

Ähnlich zur Startphase steigen die **Signalisierungs- und Speicherkosten während der Betriebsphase** von HRM. Auch hier wurde überwiegend lineares Verhalten für den verursachten Signalisierungs- und Speicheraufwand festgestellt. Die Maschentopologie ist auch hier aufgrund der quadratischen Zunahme von Links bzw. Pfaden mit steigender Knotenanzahl die Ausnahme, sodass die auftretenden Signalisierungen sowie der Speicheraufwand quadratisch zunehmen. Allgemein gilt für die durch HRM verursachten Kosten:

- Die *RouteReport/RouteShare*-Nachrichten zur Verteilung von Routingdaten sind bei sehr hoher Netzdynamik maßgebend für die Gesamtkosten. Durch Senkung der Signalisierungsrate können die verursachten Signalisierungsdaten unabhängig der Netztopologie und anderer Konfigurationsparameter signifikant gesenkt werden[30].
- Die Menge der *AnnounceCoordinator*-Nachrichten ist wiederum maßgebend für die minimal auftretenden Signalisierungskosten, welche bei konstanten QoS-Eigenschaften (aller Routen) verursacht werden.
- Als Clusterradius sollte stets ein Wert zwischen 0 und dem Durchmesser des jeweiligen Netzwerks verwendet werden, ein Wert von 0 ist nicht empfehlenswert. Des Weiteren ist eine Anpassung des Wertes in Abhängigkeit vom Durchmesser des Netzwerks sinnvoll (auch wenn dies für die Funktionsweise von HRM nicht zwingend erforderlich ist).
- Durch die bei HRM angewandte Netzwerkunterteilung und Topologieaggregation werden der verursachte Signalisierungs- und Speicheraufwand zusätzlich reduziert.
- Mit zunehmender Hierarchietiefe fallen die Kosten geringer aus. Dabei ist jedoch die maximal mögliche Verzögerung für die Verteilung von neuen Routingdaten im Netzwerk zu beachten – sie steigt mit zunehmender Hierarchietiefe ebenfalls an.

Die Untersuchungen zur **Nutzbarkeit von Netzwerkressourcen** beim Einsatz von HRM und seiner Datenebene ergaben, dass der Vorteil von HRM im Vergleich zu BE-Routing maßgebend durch den verwendeten Clusterradius sowie der Einbeziehung von fremden Clustern beim Routing beeinflusst wird. Je größer die resultierenden Cluster ausfallen, desto mehr Details über die Topologie werden bei der Verteilung von Routingdaten abstrahiert. Werden in dem Fall fremde Cluster bei Routingentscheidungen nicht einbezogen, kann dadurch der durch HRM (im Vergleich zu BE-Routing) erzielte Gewinn (die zusätzlich zur Verfügung stehenden Netzwerkressourcen) vermindert ausfallen. Dies kann jedoch durch die Signalisierung von sogenannten Routingschleifen vermieden werden, sodass beim Routing alle Cluster einbezogen werden und somit auch alle Netzwerkressourcen durch eine Anwendung eingesetzt werden können.

Durch eine adaptierbare Filterung von *AnnounceCoordinator*-Nachrichten können mit HRM auch sogenannte **Netzwerkrichtlinien** umgesetzt werden, sodass die durch den Netzwerkbetreiber gewünschten Bedingungen beim Routing beachtet werden. Des Weiteren wurde die **Auswirkung von Topologieaggregationen** auf die resultierenden Routingentscheidungen an ausgewählten Beispielen gezeigt. Dabei wurde deutlich, dass die Aggregationsmechanismen während der Verteilung von Routingdaten Auswirkungen auf das Verhalten beim Routing haben und bei der Festlegung von Netzwerkrichtlinien beachtet werden müssen.

[30] In Kapitel 6 wurden sehr kurze Signalisierungsintervalle verwendet, die in realen Anwendungen von HRM eher geringer ausfallen können, sodass geringere Signalisierungskosten auftreten.

7 Zusammenfassung

Aktuelle Routingprotokolle, wie OSPF oder BGP, verwenden eine vorgegebene Konfiguration zur Gruppierung von Knoten als auch zur Vergabe von Adressen. Diese Einstellungen müssen durch den Netzwerkbetreiber explizit vorgegeben werden. Die Vermeidung dieses administrativen Mehraufwands war das vordergründige Ziel bei der Konzeption von HRM, wodurch sich die vorliegende Arbeit insbesondere von alternativen Ansätzen unterscheidet und dem Routingmanagement seinen innovativen Charakter verleiht. Zu den sieben wichtigsten Vorteilen von HRM gegenüber Alternativlösungen zählen:

- **Autonome Unterteilung des Netzwerks sowie Adressvergabe**: Die Signalisierungen der drei in dieser Arbeit entwickelten Protokolle von HRM laufen ausschließlich autonom ab, wobei die dafür notwendigen Parameter (z.B. Signalisierungsintervalle, Clusterradius, Hierarchiehöhe) global festgelegt sind und bei Bedarf zusätzlich angepasst werden können. Durch das **erste Protokoll** wird das Netzwerk in Abschnitte (Cluster) unterteilt und eine Hierarchie aus Kontrollinstanzen erstellt. Das **zweite Protokoll** verwendet diese Hierarchie, um jeder Netzwerkschnittstelle eine global eindeutige Adresse (HRMID) zuzuweisen. Auf der Basis der dadurch festgelegten Strukturierung des Netzwerks und der verteilten Adressen werden durch das **dritte Protokoll** kontinuierlich Routingdaten im Netzwerk auf effiziente Art verteilt, sodass jeder Knoten seine eigene lokale Routingtabelle aufbauen und stetig aktualisieren kann. Dadurch kennt jeder Knoten in einem HRM-basierten Netzwerk automatisch alle Daten, welche für eine Routingentscheidung notwendig sind. Dies unterscheidet HRM grundsätzlich von OSPF und BGP, da diese vordefinierte Knotengruppen und bereits zugewiesene IP-Adressen für ihre Signalisierungen benötigen und erst dann ein Routing im Netzwerk ermöglichen.
- **Beachtung von Qualitätsanforderungen beim Routing:** Die Datenebene von HRM verwendet ein *Hop-by-Hop*-Routing, welches die Qualitätsanforderungen der Anwendung an die gewünschte Übertragung bei ihren Entscheidungen beachtet. Dadurch werden ungeeignete Pfade im Rahmen der vorhandenen Möglichkeiten vermieden und Engpässe bei der Übertragung von Anwendungsdaten verhindert. Somit wird die Dienstqualität von Anwendung unterstützt und im Kontext von *IntServ* möglichst viele erfolgreiche Verbindungsanfragen ermöglicht. Dies unterscheidet HRM von dem herkömmlichen BE-Routing in heutigen IP-basierten Netzwerken.
- **Beachtung von aktuellen QoS-spezifischen Routeneigenschaften beim Routing:** Durch die Kontrollebene werden im Netzwerk kontinuierlich Routingdaten mit QoS-spezifischen Eigenschaften zu jeder bekannten Route verteilt. Diese Daten werden von jedem Knoten zum Aufbau und zur automatischen Aktualisierung einer lokalen Routingtabelle verwendet. Dadurch können bei Routingentscheidungen der Datenebene die aktuellen QoS-spezifischen Eigenschaften von allen Routen zum gewünschten Ziel beachtet werden. Dies unterscheidet HRM von heutigem BE-Routing.
- **Fairness im Routing und Ausnutzung von Netzwerkressourcen:** Der Routingalgorithmus der Datenebene wendet eine Kombination aus *Widest Shortest Path Routing* (WSPF) und *Shortest Widest Path Routing* (SWPF) an. Dadurch wird die kürzeste Route zum Ziel bevorzugt, bis diese die Qualitätsanforderungen der Anwendungen nicht mehr erfüllt. Dadurch werden vorranging möglichst wenige parallel ablaufende Routinganfragen beeinflusst oder gar blockiert. Sollte die WSPF-Route nicht für die Übertragung der Anwendungsdaten entsprechend der vorgegebenen Anforderungen genügen oder zu stark ausgelastet sein, wird auf SWPF-Routing umgeschaltet und jene Route mit der größten Kapazität verwendet. Durch dieses Vorgehen werden letztlich alle im Netzwerk vorhandenen Routen beachtet und die Ressourcen nach und nach ausgenutzt. Das ist insbesondere bei der Übertragung von audiovisuellen Datenströmen vorteilhaft, da hierbei eine möglichst gute Wiedergabe in Echtzeit auf Empfängerseite notwendig ist, um eine möglichst gute Dienstqualität gegenüber dem Nutzer zu erbringen.

- **Eigenständigkeit:** Der fünfte besondere Vorteil von HRM ist die erreichte Unabhängigkeit gegenüber vorhandenen Protokollen von Schicht 3 des OSI-Modells. Zu diesem Zweck wird für die Signalisierungen der Kontrollebene eine Adressierung für Knoten auf Basis von sogenannten Knoten-IDs verwendet, die knotenlokal verwaltet werden können. Dadurch verbleiben die Abläufe der Kontrollebene unabhängig von anderen Protokollen und benötigen für ihre Signalisierungen insbesondere keine IP-Adressen im Netzwerk.
- **Kompatibilität:** Durch ihre Eigenständigkeit sind die Abläufe der Kontrollebene sowohl für IPv4- als auch IPv6-basierte Netzwerke anwendbar. Des Weiteren ist das eingesetzte Adressierungsschema der HRMIDs kompatibel zu IP-Adressen, sodass das durch die Datenebene von HRM bereitgestellte QoS-Routing auch für heutige Netzwerke einsetzbar ist. Details zu dieser Interoperabilität sind in den Abschnitten 3.4.5 und 3.9 zu finden. Zusätzlich kann das Konzept von HRM ohne Anpassungen direkt in FoG-basierten Netzwerken eingesetzt werden. Dieser Gedanke ist in Kapitel 4 aufgegriffen und durch eine vollständige Implementierung des Routingmanagements untermauert. HRM eignet sich – neben dem Einsatz in heutigen Netzwerken – auch als sinnvolle Erweiterung für FoG in Form eines eigenständigen Routingdienstes.
- **Skalierbarkeit:** Für eine gute Skalierbarkeit wird das Netzwerk durch die Kontrollebene automatisch in Cluster aufgeteilt. Dadurch wird die Basis für die Verteilung von Speicheraufwand und Berechnungslast zwischen den Knoten im Netzwerk geliefert. Da für ein solches Netzwerkmanagement (ohne die Nutzung externer Topologiedatenbanken) Hallo-Nachrichten zur Erkennung von Nachbarschaftsbeziehung unumgänglich sind – häufig kommen dabei Broadcast-Nachrichten zum Einsatz – muss ihre Anzahl möglichst gering gehalten werden. Zu diesem Zweck werden die Cluster mit Hilfe einer mehrstufigen Hierarchie von Koordinatorinstanzen verwaltet, sodass Hallo-Nachrichten (z.B. *AnnounceNeighborNode, AnnounceCoordinator*) in Abhängigkeit vom Hierarchielevel möglichst nur mit begrenzter Reichweite weitergeleitet und dabei auf den Clusterradius beschränkt werden. Die Hierarchie der Kontrollebene wird ebenfalls für die Verteilung von Routingdaten verwendet. Dabei wird durch Topologieaggregation der verursachte Signalisierungsaufwand zusätzlich reduziert.

Durch diese Vorteile ermöglicht HRM im Vergleich zu heutigen OSPF-oder BGP-basierten Lösungen geringere Kosten beim Aufbau und der Wartung von Netzwerken. Zusätzlich ermöglicht das neue Routingmanagement im Gegensatz zu reinem BE-basiertem Routing eine bessere Ausnutzung von vorhandenen Ressourcen im Netzwerk.

Während bei der Konzeption vor allem Firmennetzwerke im Vordergrund standen, ist ebenfalls eine Anwendung für sehr große Providernetzwerke möglich. Topologieänderungen im Netzwerk können einen Umbau innerhalb der Managementinfrastruktur auslösen, wodurch die Kommunikation temporär gestört werden kann. Das Konzept ist daher mit Einschränkungen für sehr hochdynamische adhoc-Netzwerke geeignet und sollte für Topologien mit eher gleichbleibender Konnektivität der Knoten eingesetzt werden.

In dieser Arbeit wurde ebenfalls eine Implementierung von HRM vorgestellt. Diese wurde in den FoG-SiEm-Netzwerksimulator integriert, wodurch das neue Routingmanagement sowohl für IP- als auch FoG-basierte Netzwerk untersucht werden kann. Der dabei entstandene Quellcode wurde der Öffentlichkeit als Open-Source-Lösung zur Verfügung gestellt, sodass weiterführende Experimente auf der Basis des aktuellen Kenntnisstands möglich sind. Innerhalb dieser Arbeit wurden die Möglichkeiten der Implementierung zur Einschätzung der Kosten sowie des erreichten Nutzens von HRM verwendet. Bei den durchgeführten Messungen wurde deutlich, dass die Signalisierungskomplexität während der Betriebsphase eines Netzwerks mit steigender Knotenanzahl weitgehend linear ansteigt. Die Ausnahme bildet dabei die Maschentopologie, welche aufgrund der quadratischen Zunahme von Link bzw. Routen auch zu einer quadratischen Zunahme der Signalisierungen führt.

Als zusätzliches Anschauungs- und Versuchsobjekt entstand im Rahmen dieser Arbeit die eigenständige Videokonferenzsoftware Homer-Conferencing (kurz: Homer). Die Vorteile der Software im wissenschaftlichen Kontext (insbesondere zur Evaluierung von Routing- und Transportsystemen) sind:

- **Explizite Generierung und Übertragung von audiovisuellen Datenströmen:**
  - Durch seine zahlreichen **Konfigurationsmöglichkeiten** eignet sich Homer insbesondere für die Untersuchung von Übertragungsqualitäten. Dabei können senderseitig die Eigenschaften jedes Stroms, zum Beispiel der verwendete Codec oder die Bildauflösung, mit Hilfe von grafischen Dialogen eingestellt werden. Dadurch unterscheidet sich Homer von alternativer Software.
  - Für ausgewählte audiovisuelle Datenströme erlaubt die Software über ihre grafischen Dialoge die Definition von **Qualitätsanforderungen.** Diese werden an den verwendeten Netzwerkstack übergeben, sodass sie durch das Netzwerk bei der Übertragung beachtet werden. Dies ist insbesondere für die Untersuchung von Routing unter Beachtung von anwendungsspezifischen Anforderungen notwendig und wird von alternativer Software in dieser Form nicht unterstützt.
  - Die Softwarearchitektur beinhaltet eine universelle Schnittstelle für die Verwendung von Kommunikationsfunktionen, sodass dadurch ein einfacher **Austausch des verwendeten Netzwerkstacks und** dem jeweils eingesetzten **Transportprotokoll** möglich ist. Durch dieses Design kann die Software für qualitative Vergleiche zwischen verschiedenen Routingansätzen und Protokollimplementierungen in realen Testszenarien eingesetzt werden. Diese Flexibilität ist mit alternativer Software nicht möglich.
- **Überwachung und Wiedergabe von audiovisuellen Datenströmen:**
  - Innerhalb der grafischen Oberfläche sind zusätzliche Anzeigen vorhanden, welche eine Beobachtung von ausgehenden und eingehenden audiovisuellen Datenströmen anhand von **Statistiken in Echtzeit** ermöglichen. Dadurch können insbesondere gemessene **Paketverluste** für eingehende Ströme überwacht werden, sodass dadurch die Qualität der aktuellen Übertragung bzw. des zugrundeliegenden Routings eingeschätzt werden kann. Außerdem können damit kontinuierlich die auftretenden Paketgrößen und Übertragungsverzögerungen gemessen und ausgegeben werden.
  - Parallel zur Ausgabe von Messdaten existieren **Wiedergabemöglichkeiten**, sodass jeder eintreffende Datenstrom über die lokalen Lautsprecher bzw. den Monitor in Echtzeit wiedergegeben werden kann. Die Übertragungsqualität kann somit zusätzlich anhand von akustischen bzw. visuellen Unterschieden bewertet werden.

Die Software ist als fertiges Softwarepaket jeweils für Windows, Linux und OS X öffentlich zugänglich. Ausgewählte Teile wurden zusätzlich in FoGSiEm integriert, um auch innerhalb der Netzwerksimulationssoftware eine vollständige Videostreaming-Lösung zur Verfügung zu stellen. Die resultierenden Funktionen wurden mehrfach für öffentliche Vorführungen der Konzeption und der sich daraus ergebenden Vorteile von HRM sowie FoG eingesetzt. Zusätzlich wurde ein mögliches zukünftiges Anwendungsszenario für Homer vorgestellt, in welchem die Software zur Liveübertragung eines Versuchsaufbaus sowie zur interaktiven Lehrer-Schüler-Kommunikation dient.

8 Ausblick

Während der Arbeit an HRM entstanden Ideen für weiterführende Themenstellungen. Sie werden nachfolgend im Überblick vorgestellt.

Zusätzliche Datenreduktion durch Schwellwerte für den Anstoß von Signalisierungen

Bei der Verteilung von Routingdaten werden aktuell sowohl Teil- als auch Vollaktualisierungen verwendet. Erstere werden bei Statusänderungen an bekannten Routen ausgelöst, während letztere periodisch unabhängig von Veränderungen im Netzwerk verschickt werden. Durch Einführung von Schwellwerten ist es möglich, Teilaktualisierungen ausschließlich bei umfassenderen Statusänderungen auszulösen und somit die Datenreduktion zusätzlich zu verbessern.

Neustarts von Routern

Da die Kontrollebene auf jede Veränderung an der Netzwerktopologie reagiert, sollten unnötige Link- oder Routerneustarts vermieden werden, um die Struktur der Kontrollebene über einen langen Zeitraum stabil zu halten. Dennoch ist ein Neustart unter Umständen während des Netzwerkbetriebs notwendig. In diesem Fall wird HRM nach Wiederherstellung des Routerbetriebs zur gleichen Struktur zurückfinden. Um die dabei verursachten Signalisierungen weitgehend zu vermeiden ist es denkbar, dass der schwindende Router sich explizit für eine definierte Zeit an den restlichen Entitäten der Kontrollebene abmeldet und erst nach Überschreitung einer festgelegten Zeit als „nicht mehr verfügbar“ wahrgenommen wird, sodass die Umstrukturierung der Kontrollebene in diesem Fall verzögert startet. Verschiedene ähnliche Ansätze existieren bereits unter der Bezeichnung *graceful restart* für OSPF [150] [151] und BGP [152].

Alternative Kriterien für die Koordinatorplatzierung

Für HRM werden in dieser Arbeit die Prioritäten einzelner Kandidaten für die Koordinatorwahl verwendet. Die jeweiligen Werte sind abhängig von der Konnektivität (Hierarchielevel 0) und der Anzahl und Entfernung umliegender Koordinatorinstanzen (Hierarchielevel 1 und darüber). Als weitere Kriterien für die Platzierung der Instanzen wären unter anderem denkbar: Prozessor-/Speicherkapazität der Router oder die aktuell festgelegte Netzwerkrichtlinie. Des Weiteren wäre auch eine Platzierung anhand von angebotenen Inhalten des Netzwerks denkbar, sodass dadurch die Umsetzung von sogenannten *Content Delivery Networks* unterstützt wird. HRM kann in diesem Kontext auch innerhalb eines Overlay-Netzwerks angewandt werden.

Erkennung von Netzwerkgrenzen

Trotz der gewünschten autonomen Arbeitsweise des Routingmanagements ist es wichtig, etwaige Netzwerkgrenzen konfigurieren zu können, sodass festgelegte Netzwerkrichtlinien, geografische Unterschiede (Stadtgrenzen) oder Infrastrukturunterschiede (Gebäudegrenzen) im Routing beachtet werden. Bei HRM stehen dafür die *AnnounceCoordinator*-Nachrichten zur Verfügung, deren Ausbreitung explizit begrenzt werden kann, um die Platzierung von Koordinatorinstanzen zu beeinflussen. Zusätzlich wäre es aber auch denkbar, dass HRM zusätzliche Informationen aus externen Datenbasen einbezieht, sodass Routingzonen automatisch eingerichtet werden.

Verbesserung der Topologieaggregation

Oftmals können Knoten eines Clusters über mehrere Gateways erreicht werden, sodass eine Gewichtung zwischen diesen Zugangspunkten anhand der jeweils vorhandenen Linkkapazitäten das Routing positiv beeinflussen würde. Bei der Verteilung von Routingdaten durch die Kontrollebene werden jedoch zur Beschreibung der Pfade zwischen einem Clustergateway und clusterinternen Knoten ausschließlich stark aggregierte Informationen im Netzwerk verteilt.

Integration von Sicherheitsmerkmalen

Die in dieser Arbeit vorgestellten Signalisierungen fokussieren auf der Synchronisation von Statusdaten in einer verteilten Umgebung – Sicherheitsaspekte werden dabei jedoch nicht näher betrachtet. Dennoch ist die Absicherung von Signalisierungen gegenüber der böswilligen Einstreuung von Daten aus unautorisierter Quelle ein sehr aktuelles Thema, sodass bereits Spezifikationen für DNS [153] [154] und OSPF [155] existieren. Ähnliche Mechanismen sind ebenfalls für die Signalisierungen der Kontrollebene denkbar, sodass die Aktualisierung von Routingdaten nur durch autorisierte Entitäten erfolgt.

A Konfiguration heutiger Routingprotokolle

Die folgenden Abschnitte vermitteln einen Eindruck über den Administrationsaufwand für OSPFv3 und BGPv4. Die Ausführungen erheben nicht den Anspruch auf Vollständigkeit, stattdessen werden ausgewählte Teile vorgestellt.

A.1 OSPF

OSPF benötigt folgenden zentralen Konfigurationsparameter (Auszug aus Abschnitt C.1 in [6]):

> In general, a separate copy of the OSPF protocol is run for each area. Because of this, most configuration parameters are defined on a per-area basis. The few global configuration parameters are listed below.
>
> Router ID
>
> This is a 32-bit number that uniquely identifies the router in the Autonomous System. If a router's OSPF Router ID is changed, the router's OSPF software should be restarted before the new Router ID takes effect. Before restarting due to a Router ID change, the router should flush its self-originated LSAs from the routing domain [..]. Otherwise, they will persist for up to MaxAge seconds.
>
> Because the size of the Router ID is smaller than an IPv6 address, it cannot be set to one of the router's IPv6 addresses (as is commonly done for IPv4). Possible Router ID assignment procedures for IPv6 include: a) assign the IPv6 Router ID as one of the router's IPv4 addresses or b) assign IPv6 Router IDs through some local administrative procedure (similar to procedures used by manufacturers to assign product serial numbers).
>
> The Router ID of 0.0.0.0 is reserved and SHOULD NOT be used.

Eine OSPF-Area wird durch folgende Parameter festgelegt (Auszug aus Abschnitt C.2 in [6]):

> All routers belonging to an area must agree on that area's configuration. Disagreements between two routers will lead to an inability for adjacencies to form between them, with a resulting hindrance to the flow of both routing protocol information and data traffic. The following items must be configured for an area:
>
> Area ID
>
> This is a 32-bit number that identifies the area. The Area ID of 0 is reserved for the backbone.
>
> List of address ranges
>
> Address ranges control the advertisement of routes across area boundaries. Each address range consists of the following items:
>
> [IPv6 prefix, prefix length]
>
> Describes the collection of IPv6 addresses contained in the address range.
>
> Status
>
> Set to either Advertise or DoNotAdvertise. Routing information is condensed at area boundaries. External to the area, at most a single route is advertised (via a inter-area-prefix-LSA) for each address range. The route is advertised if and only if the address range's Status is set to Advertise. Unadvertised ranges allow the existence of certain networks to be intentionally hidden from other areas. Status is set to Advertise by default.

ExternalRoutingCapability
    Whether AS-external-LSAs will be flooded into/throughout the area. If AS-external-LSAs are excluded from the area, the area is called a stub area or NSSA. Internal to stub areas, routing to external destinations will be based solely on a default inter-area route. The backbone cannot be configured as a stub or NSSA area. Also, virtual links cannot be configured through stub or NSSA areas. [..]

StubDefaultCost
    If the area has been configured as a stub area, and the router itself is an area border router, then the StubDefaultCost indicates the cost of the default inter-area-prefix-LSA that the router should advertise into the area. [..]

NSSATranslatorRole and TranslatorStabilityInterval
    [..] Additionally, an NSSA Area Border Router (ABR) is also required to allow configuration of whether or not an NSSA default route is advertised in an NSSA-LSA. If advertised, its metric and metric type are configurable. [..]

ImportSummaries
    When set to enabled, prefixes external to the area are imported into the area via the advertisement of inter-area-prefix-LSAs. When set to disabled, inter-area routes are not imported into the area. The default setting is enabled. This parameter is only valid for stub or NSSA areas.

Für die Routerschnittstelle existieren folgende Parameter (Auszug aus Abschnitt C.3 in [6]):

Some of the configurable router interface parameters (such as Area ID, HelloInterval, and RouterDeadInterval) actually imply properties of the attached links. Therefore, these parameters must be consistent across all the routers attached to that link. The parameters that must be configured for a router interface are:

IPv6 link-local address
    The IPv6 link-local address associated with this interface. May be learned through auto-configuration.

Area ID
    The OSPF area to which the attached link belongs.

Instance ID
    The OSPF protocol instance associated with this OSPF interface. Defaults to 0.

Interface ID
    32-bit number uniquely identifying this interface among the collection of this router's interfaces. For example, in some implementations it may be possible to use the MIB-II IfIndex [..].

IPv6 prefixes
    The list of IPv6 prefixes to associate with the link. These will be advertised in intra-area-prefix-LSAs.

Interface output cost(s)
    The cost of sending a packet on the interface, expressed in the link-state metric. This is advertised as the link cost for this interface in the router's router-LSA. The interface output cost MUST always be greater than 0.

```
RxmtInterval
    The number of seconds between LSA retransmissions for adjacencies be-
    longing to this interface. Also used when retransmitting Database De-
    scription and Link State Request packets. This should be well over
    the expected round-trip delay between any two routers on the attached
    link. The setting of this value should be conservative or needless
    retransmissions will result. Sample value for a local area network: 5
    seconds.

InfTransDelay
    The estimated number of seconds it takes to transmit a Link State
    pdate packet over this interface. LSAs contained in the update packet
    must have their age incremented by this amount before transmission.
    This value should take into account the transmission and propagation
    delays of the interface. It MUST be greater than 0. Sample value for
    a local area network: 1 second.

Router Priority
    An 8-bit unsigned integer. When two routers attached to a network
    both attempt to become the Designated Router, the one with the high-
    est Router Priority takes precedence. If there is still a tie, the
    router with the highest Router ID takes precedence. A router whose
    Router Priority is set to 0 is ineligible to become the Designated
    Router on the attached link. Router Priority is only configured for
    interfaces to broadcast and NBMA networks.

HelloInterval
    The length of time, in seconds, between Hello packets that the router
    sends on the interface. This value is advertised in the router's
    Hello packets. It MUST be the same for all routers attached to a com-
    mon link. The smaller the HelloInterval, the faster topological
    changes will be detected. However, more OSPF routing protocol traffic
    will ensue. Sample value for a X.25 PDN: 30 seconds. Sample value for
    a local area network (LAN): 10 seconds.

RouterDeadInterval
    After ceasing to hear a router's Hello packets, the number of seconds
    before its neighbors declare the router down. This is also advertised
    in the router's Hello packets in their RouterDeadInterval field. This
    should be some multiple of the HelloInterval (e.g., 4). This value
    again MUST be the same for all routers attached to a common link.

LinkLSASuppression
    Indicates whether or not origination of a link-LSA is suppressed. If
    set to "enabled" and the interface type is not broadcast or NBMA, the
    router will not originate a link-LSA for the link. This implies that
    other routers on the link will ascertain the router's next-hop ad-
    dress using a mechanism other than the link-LSA [..]. The default
    value is "disabled" for interface types described in this specifica-
    tion. It is implicitly "disabled" if the interface type is broadcast
    or NBMA. Future interface types MAY specify a different default.
```

Zusätzlich bietet OSPF die Funktion virtueller Links. Diese sind notwendig, insofern die *Area 0*, auch *Backbone Area* genannt, physisch unterteilt ist. Virtuelle Links fügen die Netzabschnitte in diesem Fall wiederum zu einer zusammenhängenden *Area* zusammen. Für OSPFv3 wird dazu festgelegt (Auszug aus Abschnitt C.4 in [6]):

```
[..] The virtual link appears as a point-to-point link with no global
IPv6 addresses in the graph for the backbone. The virtual link must be
configured in both of the area border routers.
```

```
[..] Virtual links do not have link-local addresses, but instead use one
of the router's global-scope IPv6 addresses as the IP source in OSPF pro-
tocol packets it sends on the virtual link. Router Priority is not used
on virtual links. Interface output cost is not configured on virtual
links, but is dynamically set to be the cost of the transit area intra-
area path between the two endpoint routers. The parameter RxmtInterval
may be configured and should be well over the expected round-trip delay
between the two routers. This may be hard to estimate for a virtual link;
it is better to err on the side of making it too long.
A virtual link is defined by the following two configurable parameters:
the Router ID of the virtual link's other endpoint and the (non-backbone)
area that the virtual link traverses (referred to as the virtual link's
transit area). [..]
```

Weitere Konfigurationsparameter sind in den Abschnitten C.5 bis C.7 in [6] zu finden:

```
C.5. NBMA Network Parameters
    [..]
C.6. Point-to-Multipoint Network Parameters
    [..]
C.7. Host Route Parameters
    [..]
```

A.2 BGP

Für einen BGP-Router, auch *BGP speaker* genannt, müssen mindestens folgende globale Einstellungen gewählt werden (Auszug aus Abschnitt 4.2 in [42]):

```
My Autonomous System:
    This 2-octet unsigned integer indicates the Autonomous System number
    of the sender.

Hold Time:
    This 2-octet unsigned integer indicates the number of seconds the
    sender proposes for the value of the Hold Timer. Upon receipt of an
    OPEN message, a BGP speaker MUST calculate the value of the Hold
    Timer by using the smaller of its configured Hold Time and the Hold
    Time received in the OPEN message. The Hold Time MUST be either zero
    or at least three seconds. An implementation MAY reject connections
    on the basis of the Hold Time. The calculated value indicates the
    maximum number of seconds that may elapse between the receipt of suc-
    cessive KEEPALIVE and/or UPDATE messages from the sender.

BGP Identifier:
    This 4-octet unsigned integer indicates the BGP Identifier of the
    sender. A given BGP speaker sets the value of its BGP Identifier to
    an IP address that is assigned to that BGP speaker. The value of the
    BGP Identifier is determined upon startup and is the same for every
    local interface and BGP peer.
```

Kommandos	Erläuterung
router bgp ***as-number***	„Enables a BGP routing process, which places the router in router configuration mode.“
network ***network-number*** **[mask** ***network-mask*****] [route-map** ***route-map-name*****]**	„Flags a network as local to this autonomous system and enters it to the BGP table.“
neighbor {*ip-address* | *peer-group-name*} remote-as ***as-number***	„Specifies a BGP neighbor.“

Tabelle A.1: Beispiel einer Kommandofolge zur Konfiguration eines BGP-Routers mit einem Nachbarn

Des Weiteren ist in [156] nachzulesen, dass umliegende BGP-Nachbarn explizit konfiguriert werden, um eine Verbindung zu ihnen zu starten. Tabelle A.1 zeigt das darin enthaltene Beispiel für eine minimale Konfiguration eines BGP-Routers. Diese kann je nach Szenario um weitere optionale Parameter verfeinert werden. Eine Referenz der vielfältigen Kommandos zur Konfiguration und Überwachung eines BGP-Routers der Firma Cisco ist in [157] zu finden.

B Inhalte und Übertragung der Signalisierungen von HRM

Dieser Anhang gibt einen Überblick über alle Daten, welche durch die drei Protokolle der Kontrollebene übertragen werden. Die Behandlung der nachfolgend aufgeführten Nachrichten wurde in der Implementierung jeweils in einer separaten Klasse gekapselt, sodass die Signalisierungen möglichst einfach nachvollzogen werden können. Auf eine bitgenaue Beschreibung wird an dieser Stelle für eine bessere Übersichtlichkeit verzichtet. Diese Details sind stattdessen der Dokumentation im Quellcode der jeweils zugehörigen Funktionen *getSerializedSize()* zu entnehmen.

B.1 Nachrichten zur Instanziierung der Kontrollebene

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
AnnounceNeighborNode		**HRM-Controller** gibt Existenz des Knoten bekannt oder antwortet auf eine fremde Bekanntgabe
	Knoten-ID des Senders	Identifikation des Knotens auf dem der sendende Koordinator instanziiert ist
	Anfrage-ID	Identifikation der Anfrage, sodass eine Zuordnung zu Antworten möglich ist
	Anfrage/Antwort-Marker	Markierung, ob es sich bei der Nachricht um eine Anfrage oder eine Antwort handelt

Tabelle B.1: Inhalt von Nachrichten zur Erkennung der Nachbarschaft

Die in Tabelle B.1 dargestellten Daten von *AnnounceNeighborNode*-Nachrichten werden zur Erkennung von Nachbarknoten benötigt und entsprechen der Beschreibung aus Abschnitt 3.6.2.1. Ähnlich Abschnitt 5.5.4 in [158] kann die Sendezeit der initialen Anfrage als Anfrage-ID genutzt werden. Dadurch ist bei Empfang einer Antwort sowohl eine Zuordnung zur Anfrage als auch eine Ermittlung der aktuellen Verzögerungszeit für den jeweiligen Link möglich. Innerhalb der Implementierung werden jedoch die FoGSiEm-spezifischen Mechanismen verwendet.

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
AnnounceCoordinator		**Koordinator** gibt seine Existenz im Netzwerk bekannt und informiert über eine Route zu ihm
	Knoten-ID des Senders	Identifikation des Knotens auf dem der sendende Koordinator instanziiert ist
	Entität-ID des Senders	Identifikation des sendenden Koordinators
	Entität-ID des letzten Hops (nur für Hierarchietiefen ab 4)	Erkennung von Clustergrenzen auf dem jeweiligen Hierarchielevel, sodass der Clusterradius während der Weiterleitung geprüft und begrenzt werden kann
	Hop-Zähler	der Wert wird bei Erreichen einer Clustergrenze um 1 erhöht, sodass er bei der Weiterleitung irgendwann den maximalen Clusterradius erreicht
	Gültigkeitsdauer	gibt die Zeitdauer an, für die der Koordinator weiterhin als gültig gilt
	Route zum Sendeknoten	enthält eine Liste eindeutiger Knoten-IDs, welche die Route zum Sender beschreibt
InvalidCoordinator		**Koordinator** gibt seine Löschung einmalig bekannt, wonach er unmittelbar entfernt wird
	Knoten-ID des Senders	Identifikation des Knotens auf dem der sendende Koordinator instanziiert ist
	Entität-ID des Senders	Identifikation des sendenden Koordinators
	Entität-ID des letzten Hops	Erkennung von Clustergrenzen auf dem jeweiligen Hierarchielevel, sodass der Clusterradius während der Weiterleitung geprüft und begrenzt werden kann

	(nur für Hierarchietiefen ab 4)	
	Hop-Zähler	der Wert wird bei Erreichen einer Clustergrenze um 1 erhöht, sodass er bei der Weiterleitung irgendwann den maximalen Clusterradius erreicht
	Route zum Sendeknoten	enthält eine Liste eindeutiger Knoten-IDs, welche die Route zum Sender beschreibt

Tabelle B.2: Inhalt von Nachrichten zur Koordinatorbekanntgabe

Tabelle B.2 stellt die Daten von *AnnounceCoordinator*- und *InvalidCoordinator*-Nachrichten dar, welche zur Bekanntgabe von Koordinatoren im Radius *r* benötigt werden. Die Darstellung entspricht den Erläuterungen von Abschnitt 3.6.2.2.

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
RequestClusterMembership		**Clustermanager** sendet eine Anfrage an einen untergeordneten Koordinator, dass dieser ein Mitglied seines Clusters werden soll
	(Knoten-ID des Senders)[1]	Identifikation des Knotens auf dem der sendende Clustermanager instanziiert ist
RequestClusterMembershipAck	-	**Koordinator** sendet positive Bestätigung an Clustermanager und gilt nachfolgend als eines seiner Clustermitglieder
InformClusterLeft	-	**Koordinator** sendet negative Bestätigung an Clustermanager oder beendet explizit seine Clustermitgliedschaft
InformClusterMembershipCanceled	-	**Clustermanager** informiert Koordinator, dass er sowie der zugehörige übergeordnete Cluster zukünftig nicht mehr zur Verfügung stehen

Tabelle B.3: Inhalt von Nachrichten zur Clustererstellung

Alle Nachrichten zur Clusterbildung werden zwischen dem Clustermanager und seinen (potentiellen) Clustermitgliedern (Koordinatoren des untergeordneten Hierarchielevels) ausgetauscht. Tabelle B.3 gibt einen Überblick über alle verwendeten Nachrichten. Ihre Übertragung geschieht als sogenannte Signalisierungsdaten des in Abschnitt 3.6.1.2 vorgestellten Nachrichtenformats für Punk-zu-Punkt-Signalisierungen.

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
ElectionPriorityUpdate	-	**Clustermanager und Koordinatoren** teilen sich gegenseitig ihre neue Priorität für Wahlvorgänge mit
ElectionWinner	-	**Clustermanager** meldet sich als Wahlgewinner gegenüber den untergeordneten Koordinatoren
ElectionResign	-	**Clustermanager** meldet sich als Wahlverlierer gegenüber den untergeordneten Koordinatoren
ElectionLeave	-	**Koordinator** deaktiviert seine Wahlmitgliedschaft bei einem übergeordneten Clustermanager
ElectionReturn	-	**Koordinator** (re-)aktiviert seine Wahlmitgliedschaft bei einem übergeordneten Clustermanager

[1] Da eine Kommunikation stets bidirektional verläuft, wird für Antwortpakete ebenfalls die Rückwärtsroute zum Clustermanager benötigt, der die Signalisierungen startete. Es erscheint daher für eine Implementierung für IP sinnvoll, innerhalb der initialen *RequestClusterMembership*-Nachricht die Knoten-ID des Senders zu speichern, sodass jeder Knoten sich mit Hilfe dieser Information und der vorhandenen Daten über Nachbarknoten eine Standardroute zum Sender ableiten kann. Diese kann solange verwendet werden, bis über eintreffende *AnnounceCoordinator*-Nachrichten eine explizit signalisierte Route gelernt wird. Die Zuordnung nachfolgender Pakete kann wiederum über die Entität-ID des Senders erfolgen. Daraus leitet sich die zusätzliche Bedingung ab, dass bei einer IP-Implementierung die Entität-ID zufällig erzeugt werden müssen, um eine möglichst eindeutige (durch eine geringe Wahrscheinlichkeit von Kollisionen) Identifikation des Senders zu ermöglichen.
Die in dieser Arbeit verwendete Implementierung verwendet jedoch die durch FoG automatisch bereitgestellte Rückwärtsroute zum Sender einer Nachricht und benötigt dieses zusätzliche Datum nicht.

ElectionAlive	-	**Clustermanager und Koordinatoren** teilen als Reaktion auf eine *PingPeer*-Nachricht ihre zukünftige Verfügbarkeit mit

Tabelle B.4: Inhalt von Nachrichten zur Koordinatorenwahl

Tabelle B.4 gibt einen Überblick über die bei Koordinatorwahlen übertragenen Nachrichten. Es ist zu erkennen, dass dabei keine zusätzlichen Signalisierungsdaten benötigt werden. Für den Ablauf der Wahlvorgänge sind ausschließlich die Nachrichtentypen wichtig. Entsprechend Abschnitt 3.6.1.3 wird dabei in jeder Signalisierung die Priorität des Senders mitübertragen, sodass eine schnelle Konvergenz der Wahlvorgänge unterstützt wird.

B.2 Nachrichten zur Adresszuweisung

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
AssignHRMID		**Koordinator** weist einem untergeordneten Clustermitglied eine Adresse zu
	HRMID	Adresse für das jeweilige Clustermitglied
	Firm-Marker	Markierung, ob die übermittelte Adresse bindend ist oder eine Anfrage nach einer ehemals zugewiesenen Adresse erlaubt ist
RequestHRMID		**Clustermitglied** stellt Anfrage nach ehemals zugewiesener Adresse an übergeordneten Koordinator
	HRMID	alte Adresse

Tabelle B.5: Inhalt von Nachrichten zur Adresszuweisung

Tabelle B.5 gibt einen Überblick über die Nachrichten der Kontrollebene, welche zur Adressverteilung innerhalb des Netzwerks eingesetzt werden. Die aufgeführten Signalisierungsdaten werden dabei als Signalisierungsdaten des in Abschnitt 3.6.1.2 vorgestellten Nachrichtenformats für Punk-zu-Punkt-Signalisierungen übertragen.

B.3 Nachrichten zur Verteilung von Routingdaten

Nachrichtename (Signalisierungstyp)	**Zusätzliche Signalisierungsdaten**	**Bedeutung**
AnnounceHRMIDs		**L0-Clustermanager** teilt die für die lokalen Netzwerkschnittstellen des Knotens zugewiesenen HRMIDs einem anderen L0-Clustermanager mit
	HRMIDs	Liste der zugewiesenen Adressen der lokalen Netzwerkschnittstellen (der Liste geht eine Längenangabe voraus)
RouteReport		**Koordinator** teilt einem übergeordneten Koordinator ausgewählte Routingdaten mit
	Routingtabelle	Liste mit Routingeinträgen
RouteShare		**Koordinator** teilt einem untergeordneten Koordinator ausgewählte Routingdaten mit
	Routingtabelle	Liste mit Routingeinträgen

Tabelle B.6: Inhalt von Nachrichten zur Signalisierung von Routingdaten

In Tabelle B.6 sind die Nachrichten aufgeführt, welche innerhalb der Kontrollebene zur Verteilung von Routingdaten eingesetzt werden. Die darin dargestellten zusätzlichen Signalisierungsdaten werden als Signalisierungsdaten des in Abschnitt 3.6.1.2 vorgestellten Nachrichtenformats für Punk-zu-Punkt-Signalisierungen übertragen.

Element	Bedeutung
Markierungen	Markierung, ob es eine Voll- oder Teilaktualisierung ist
Gültigkeitsdauer	gibt die Zeitdauer an, für die die Einträge der Tabelle als gültig gelten
Tabellenlänge	Längenangabe über die gesamte Tabelle
Einträge	Liste mit Tabelleneinträgen

Tabelle B.7: Format von übermittelten Routingtabellen

Sowohl in *RouteReport*- als auch *RouteShare*-Nachrichten werden Routingtabellen entsprechend des in Tabelle B.7 dargestellten Formats übertragen. Dabei wird eine zusätzliche Längenangabe zur Kennzeichnung der Anzahl nachfolgender Tabelleneinträge verwendet.

Element	Bedeutung
Ziel	HRMID des Ziels der Route: entweder ein Knoten oder ein Cluster aus Knoten
Quelle	HRMID des Starts der Route: ein Endknoten oder ein Gateway eines Clusters
Nächster Hop	HRMID des nächsten Knotens: ein Router oder der Endknoten der Route selbst
Hop-Zähler	Anzahl von Knoten bis zum Ziel
Datenrate	maximale verfügbare Datenrate entlang der Route
Verzögerung	minimal zu erwartende Verzögerung entlang der Route
Auslastung	maximale Auslastung, welche entlang der Route vorkommt

Tabelle B.8: Format eines Eintrages in einer übermittelten Routingtabelle

Tabelle B.8 zeigt das Format eines Eintrages einer übermittelten Routingtabelle.

B.4 Übertragung mit Hilfe von Ethernet Frames

Die Implementierung verwendet das FoG-Protokoll zur Übermittlung der Signalisierungen der Kontrollebene. Alternativ können sie auch parallel zu IPv4/v6 mit Hilfe eines vorhandenen Protokolls der Schicht 2 des OSI-Modells übertragen werden. Dafür können beispielsweise *Ethernet Frames* eingesetzt werden, um die Nachrichten von Knoten zu Knoten zu transportieren.

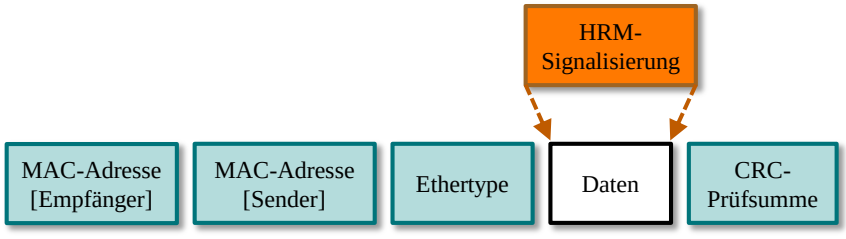

Abbildung B.1: Transport von Signalisierungsdaten der Kontrollebene mit Hilfe von *Ethernet Frames*

Abbildung B.1 zeigt den resultierenden Aufbau[2] einer Signalisierung. Entsprechend der Spezifikation von Ethernet sind allen Netzwerkschnittstellen eindeutige MAC-Adressen durch den Hardwarehersteller zugeordnet. Diese werden in Ethernet Frames sowohl für die Beschreibung des Ziels als auch der Quelle verwendet und sind aufgrund der Nachbarschaftserkennung durch *AnnounceNeighborNode-*

[2] ohne IEEE 802.1Q VLAN Tagging

Nachrichten zwischen den Nachbarknoten bekannt. Ausschließlich der in Abbildung 4.4 dargestellte Wert für *Ethertype* muss explizit für die Signalisierungen von HRM festgelegt werden. Dafür ist eine Unterscheidung zwischen drei voneinander unabhängigen Signalisierungstypen sinnvoll:

- AnnounceNeighborNode-Nachrichten zur Nachbarschaftserkennung
- AnnounceCoordinator/InvalidCoordinator-Nachrichten zur Bekanntgabe von Koordinatoren
- Sonstige Nachrichten für Punkt-zu-Punkt-Signalisierungen der Kontrollebene

Dabei wird jeweils einer der verfügbaren Werte für das Feld *Ethertype* [22] verwendet:

- **0x6070 für *AnnounceNeighborNode*-Nachrichten:** Die MAC-Adresse des Ziels sollte im Fall einer Anfrage der Broadcast-Adresse „FF:FF:FF:FF:FF:FF“ zum Erreichen aller Knoten der jeweiligen Broadcast-Domäne entsprechen. Für die Beantwortung sollte jedoch die MAC-Adresse des Senders der Anfrage als Zielidentifikation verwendet werden.
- **0x6071 für *AnnounceCoordinator*-Nachrichten:** Für diesen Nachrichtentyp werden Übertragungen zwischen verschiedenen Nachbarknoten kombiniert, sodass eine Punkt-zu-Mehrpunkt-Signalisierung realisiert wird. Dabei kommen die MAC-Adressen entsprechend der Herstellervorgaben zum Einsatz, sodass die Broadcast-Adresse hierfür nicht notwendig ist.
- **0x6072 für *InvalidCoordinator*-Nachrichten:** Dieser Nachrichtentyp wird analog zu *AnnounceCoordinator*-Nachrichten behandelt.
- **0x6073 für jegliche Punkt-zu-Punkt-Signalisierungen:** Hierbei handelt es sich um explizite Übertragungen zwischen zwei Entitäten der Kontrollebene, welche sich auf unterschiedlichen Knoten befinden können. Zur Identifikation der Quelle und des Ziels werden dabei die jeweils eindeutigen MAC-Adressen der zugehörigen Netzwerkschnittstellen verwendet.

Anwendungsdaten werden dennoch unabhängig von der zuvor beschriebenen Vorgehensweise auf Basis der herkömmlichen Mechanismen und Formate der jeweils verwendeten Protokollimplementierung für Schicht 3 übertragen. Das können sowohl IPv4/v6- als auch FoG-Paketen sein.

C Kosten der Betriebsphase von HRM für eine größere Ringtopologie

In Kapitel 6 wurden die wichtigsten empirisch ermittelten Ergebnisse zur Evaluierung von HRM vorgestellt, unzählige weitere Topologien mit verschiedenen Netzwerkgrößen sind denkbar. Dieser Anhang geht detaillierter auf eine größere Ringtopologie mit 48 Knoten ein und zeigt die auftretenden Signalisierungskosten. Der Radius wird dabei zwischen den Werten 0 und 48 variiert, die Hierarchietiefe wurde sowohl mit einem Wert von 3 als auch 4 festgelegt. Dadurch wird insbesondere der Einfluss der gewählten Hierarchietiefe auf die im Netzwerk entstehenden Signalisierungen deutlich.

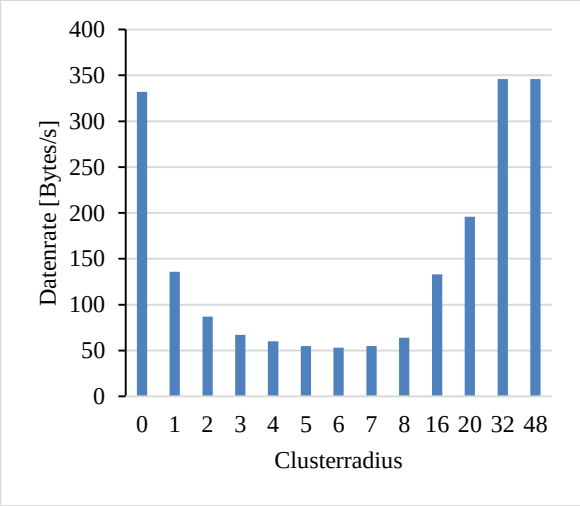

Abbildung C.1: Minimale Signalisierungskosten mit Hierarchietiefe 3 (Ring-48)

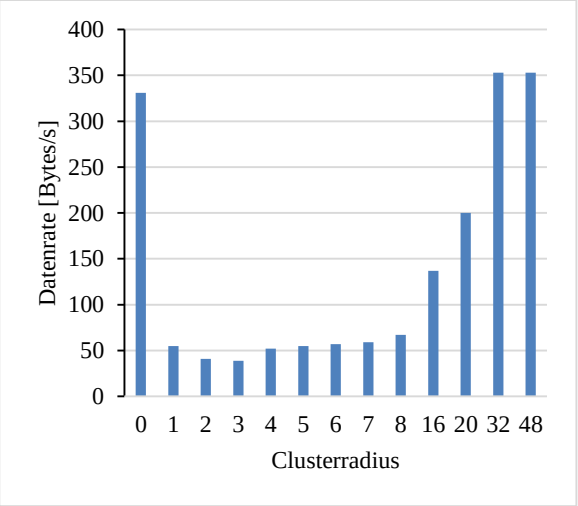

Abbildung C.2: Minimale Signalisierungskosten mit Hierarchietiefe 4 (Ring-48)

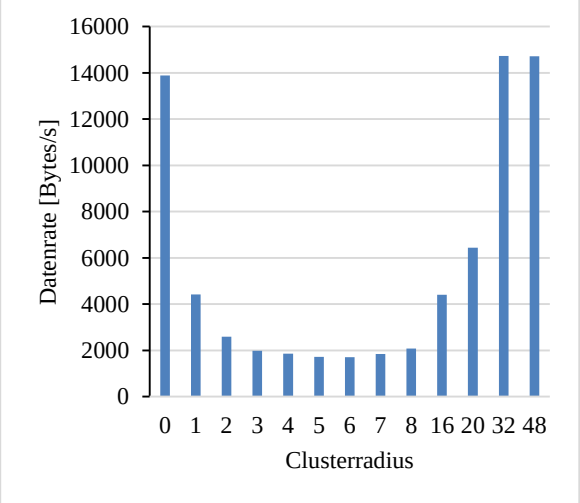

Abbildung C.3: Maximale Signalisierungskosten mit Hierarchietiefe 3 (Ring-48)

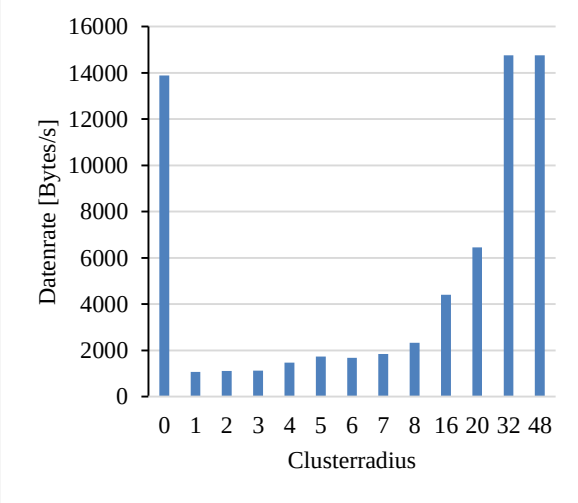

Abbildung C.4: Maximale Signalisierungskosten mit Hierarchietiefe 4 (Ring-48)

Aus den Abbildungen wird deutlich, dass für eine Hierarchietiefe von 3 ein Clusterradius von 6 optimal für möglichst geringe Datenraten der auftretenden Signalisierungen ist, während der optimale Wert für die größere Hierarchietiefe von 4 eher kleiner ausfällt. Für beide Tiefen steht ein Clusterradius von 0 und sehr große Clusterradien als ungünstige Werte fest, wodurch ein eher hohes Signalisierungsaufkommen verursacht wird.

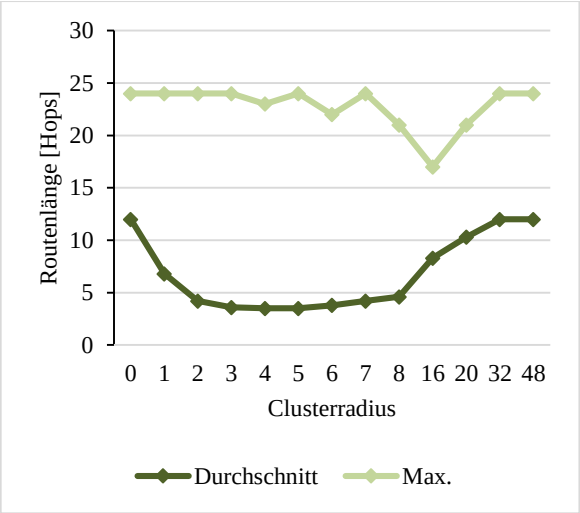

Abbildung C.5: Routenlänge zwischen den Entitäten der Kontrollebene mit Hierarchietiefe 3 (Ring-48)

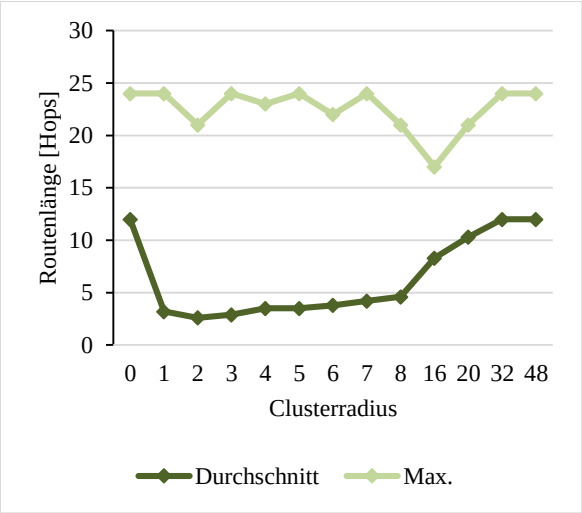

Abbildung C.6: Routenlänge zwischen den Entitäten der Kontrollebene mit Hierarchietiefe 4 (Ring-48)

Aus den ermittelten Routinglängen in Abbildung C.5 und Abbildung C.6 ist ersichtlich, dass bei optimaler durchschnittlicher Datenrate auch die Routenlängen der Signalisierungswege entsprechend gering ausfallen.

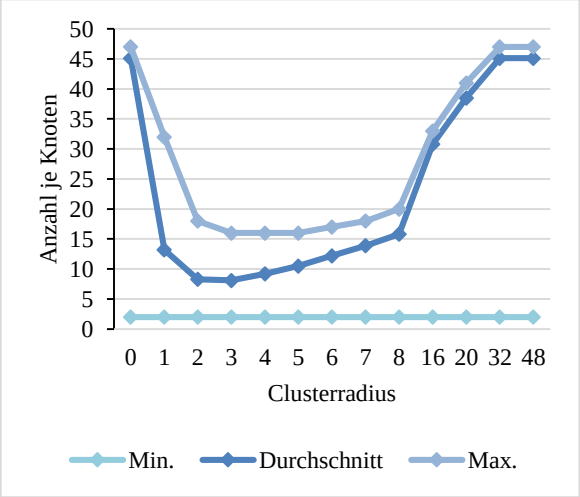

Abbildung C.7: Anzahl notwendiger Verbindungen der Kontrollebene mit Hierarchietiefe 3 (Ring-48)

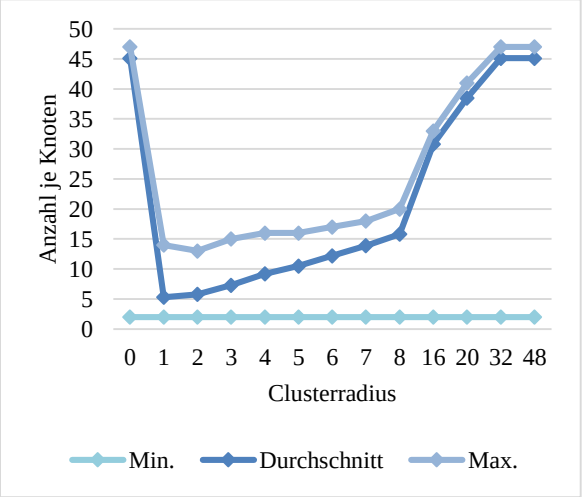

Abbildung C.8: Anzahl notwendiger Verbindungen der Kontrollebene mit Hierarchietiefe 4 (Ring-48)

Die verursachte Datenrate der Signalisierungen korreliert jedoch nicht mit der Anzahl von notwendigen Verbindungen pro Knoten. Abbildung C.7 und Abbildung C.8 zeigen, dass im Gegensatz zu vorher ein Clusterradius von 3 für eine Hierarchietiefe von 3 optimal erscheint, um die Belastung einzelner Knoten optimal zu halten.

D Grafische Dialoge der HRM-Implementierung

Dieser Anhang zeigt die grafischen Ausgabe- und Einstellmöglichkeiten der HRM-Implementierung anhand des Referenznetzwerks von Kapitel 3, eine Ringtopologie aus 8 Knoten und Links. Da die Vergabe der HRMIDs von der Reihenfolge der Speicherung untergeordneter Koordinatoren abhängt, unterscheiden sich die nachfolgende Darstellung geringfügig von Kapitel 3.

D.1 Beobachtung und Steuerung der Simulation

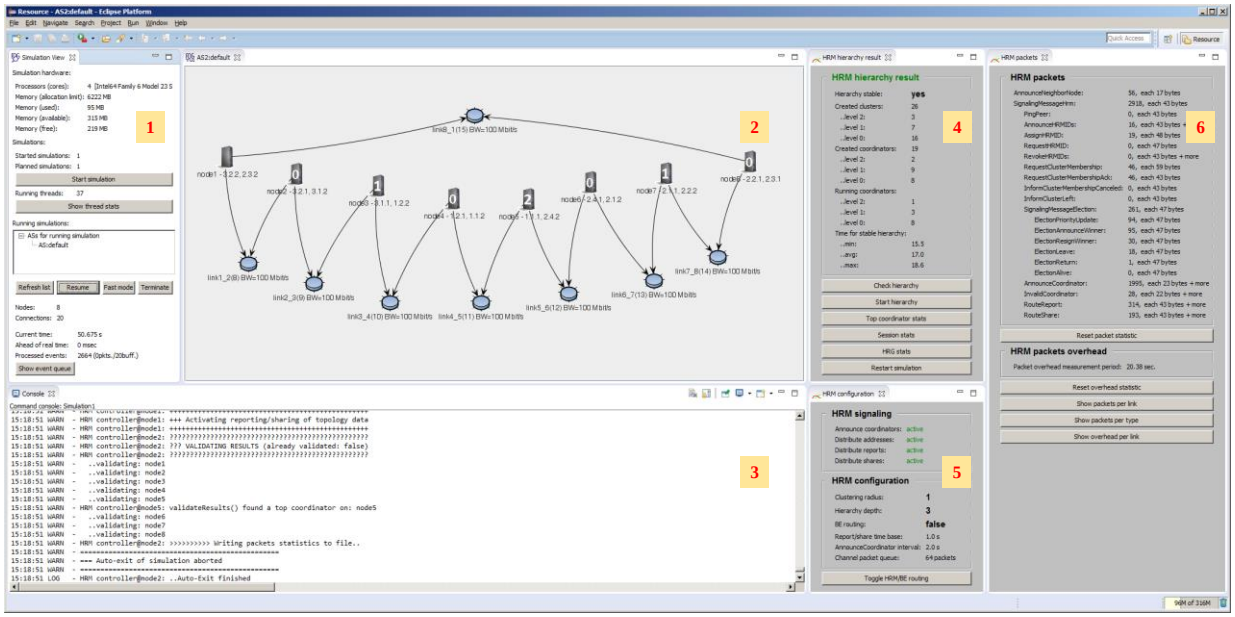

Abbildung D.1: Ausgabemöglichkeiten zur Überwachung der Abläufe

In Abbildung D.1 ist ein Experiment im Durchlauf zu sehen, dabei stehen die folgenden grafischen Ausgabefenster zur Beobachtung der Simulation zur Verfügung:

1. **Simulationsstatus**: Hier befinden sich allgemeine Steuerelemente für die Simulation sowie Statusanzeigen über Speicherverbrauch sowie die Anzahl von erstellten Verbindungen im Netzwerk.
2. **Netzwerk**: Das aktuell simulierte Netzwerk wird mit allen Knoten und Links dargestellt.
3. **Logging**: Textnachrichten werden an dieser Stelle fortlaufend ausgegeben, die eintretende Ereignisse oder Reaktionen beschreiben. Diese Ansicht dient vor allem zur Fehleranalyse.
4. **HRM-Hierarchieergebnis**: An dieser Stelle wird eine Statistik über die erstellten und den tatsächlich, bei Erreichen der finalen Lösung während der Startphase, verwendeten Entitäten der Kontrollebene angezeigt.
5. **HRM-Signalisierung/Konfiguration:** Sowohl eine Statusanzeige zu laufenden Signalisierungsabläufen als auch globale Parameter von HRM sind hier zu sehen. Erstere geben darüber Auskunft, ob die Adresszuweisung oder die Verteilung von Routingdaten bereit gestartet worden, letztere beinhalten beispielsweise den aktuellen Clusterradius oder die Hierarchietiefe. Des Weiteren kann über dieses Fenster zwischen HRM- und BE-basiertem Routing umgeschaltet werden.
6. **HRM-Pakete:** Für jeden verwendeten Nachrichtentyp wird angezeigt, wie häufig er bereits im Netzwerk für Signalisierungen verwendet wurde und wie groß eine solche Nachricht ist. Des Weiteren gibt es die Möglichkeit, weitere Statistiken zur Messung des Signalisierungsaufkommens zu steuern und auszugeben.

Mit Hilfe der aufgeführten Ausgaben und den integrierten Kontrollelementen wurden alle in dieser Arbeit vorgestellten Versuche durchgeführt bzw. gesteuert, dabei wurden ebenfalls die notwendigen Daten für die zugehörigen Messgrafiken aufgezeichnet.

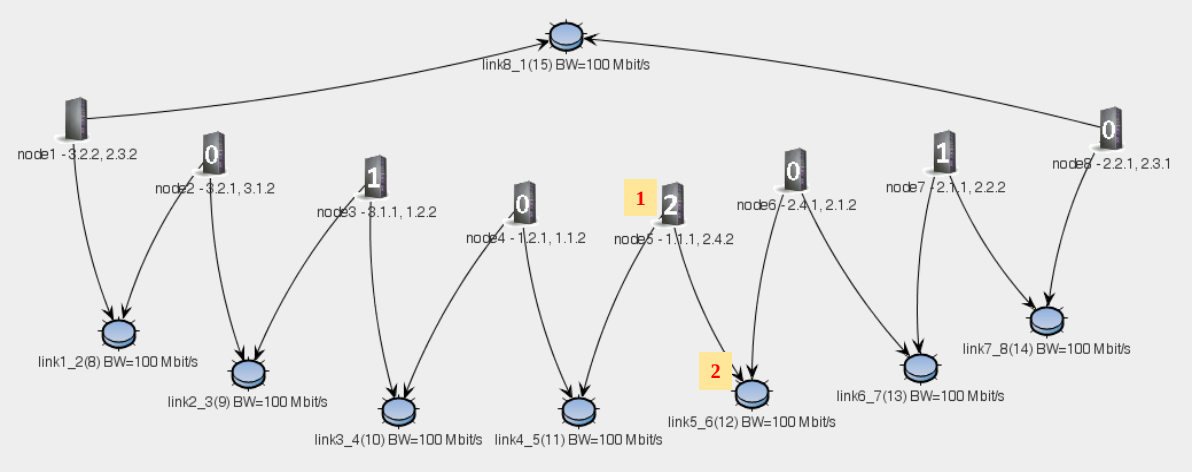

Abbildung D.2: Darstellung des Netzwerks

In Abbildung D.2 ist die Darstellung des Netzwerks nochmal detaillierter abgebildet, es sind die Knoten Links sowie zwei Markierungen zu sehen:

1. Die Zahlen auf den Knotensymbolen geben Auskunft über die Platzierung der Koordinatorinstanzen. Dabei steht die „2“ auf Knoten 5 für einen L2-Koordinator, dem TOP-Koordinator des Netzwerks, dem knotenlokal mindestens ein L1- und ein L0-Koordinator ungeordnet sind. Ähnlich verhält es sich mit Knoten 3 und 7, welche beide sowohl einen lokalen L1- als auch einen L0-Koordinator haben. Zusätzlich zu diesen Informationen sind die pro Knoten vergebenen HRMIDs jeweils unter den Symbolen aufgelistet, sodass man während der Betriebsphase stets die vergebenen Adressen im Netzwerk überwachen kann.
2. An jedem Link stehen seine Eigenschaften, wodurch das Nachvollziehen von Routingentscheidungen erleichtert wird.

Weitere Ausgabemöglichkeiten sind über Kontextmenüs erreichbar, welche ebenfalls das Starten von simulierten Anwendungen erlauben. Dazu zählen die Anwendungen *HRMTestApp* und *QoSTestApp* sowie die für die Videoübertragungsstrecke notwendigen Softwareinstanzen.

D.2 Visualisierung der Routinggraphen

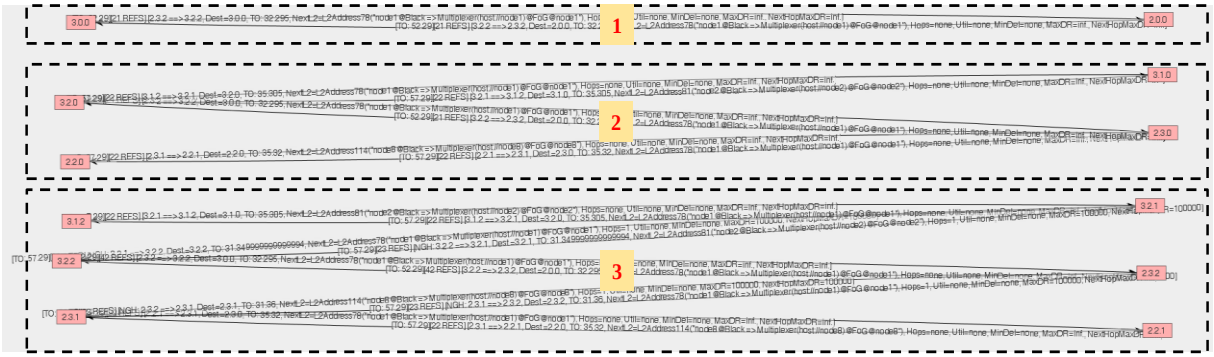

Abbildung D.3: Beispiel eines "Hierarchical Routing Graph"

Jeder Knoten besitzt lokal eine Instanz eines HRG, Abbildung D.3 zeigt den Graphen auf Knoten 1 des Szenarios. Er enthält die bekannten Routen für:

1. Hierarchielevel 1,
2. Hierarchielevel 0 sowie
3. die Links zwischen den Netzwerkschnittstellen.

Auf Basis dieser Pfadbeschreibungen können die für *RouteReport/RouteShare*-Nachrichten notwendigen Einträge der übermittelten Routingtabelle berechnet werden.

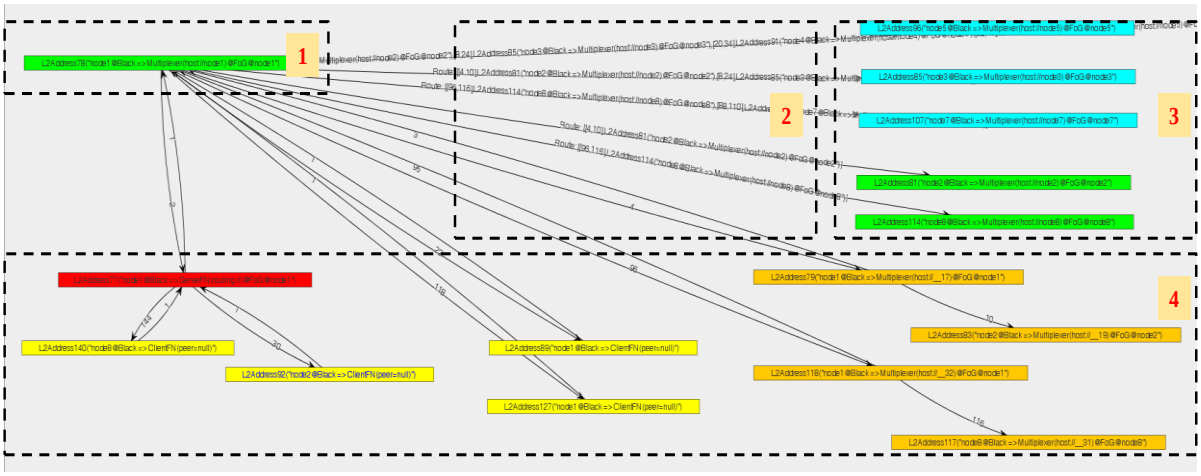

Abbildung D.4: Beispiel eines „Neighbor Routing Graph“

Zusätzlich zu *AnnounceNeighborNode*-Nachrichten dienen auch *AnnounceCoordinator*-Signalisierungen als Quelle für Einträge des NRGs. Dies wird insbesondere dazu verwendet, um eine Route zu einem entfernten Knoten zu speichern, auf dem sich eine Koordinatorinstanz befindet. In diesem Fall ist der jeweilige Zielknoten häufig nur über mindestens einen Zwischenknoten erreichbar und die Route besteht aus einer Verkettung von FoG-relevanten Routingdaten. Abbildung D.4 zeigt den resultierenden NRG für Knoten 1. Die markierten Bereiche beinhalten Daten über:

1. den zentralen FoG-Weiterleitungsknoten,
2. die bekannte kürzeste Route zu einem entfernten Knoten,
3. die Identifikation von entfernten Knoten mit bekannter Koordinatorinstanz sowie
4. die weiteren lokalen FoG-Weiterleitungsknoten.

Mit Hilfe dieser Daten kann ein Knoten Signalisierungsnachrichten in Richtung ihres jeweiligen Ziels weiterleiten.

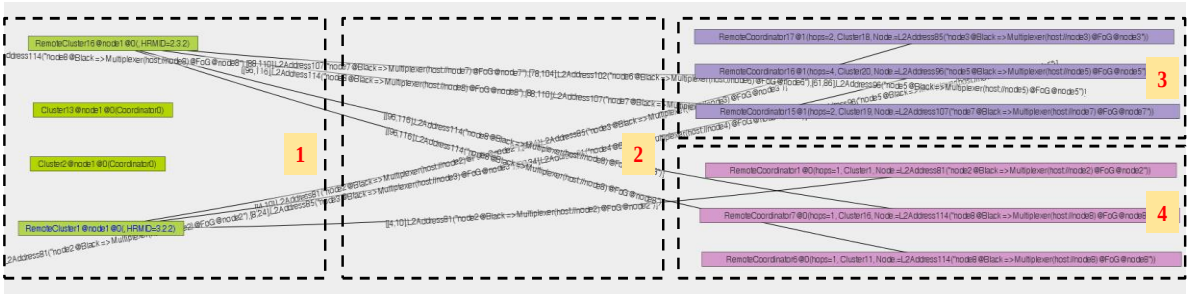

Abbildung D.5: Beispiel eines "Abstract Routing Graph"

Insbesondere während der Entwicklungsphase war es wichtig, gute Ausgabemöglichkeiten für die knotenlokalen Daten nutzen zu können. Zu diesem Zweck wurde unabhängig vom Konzept der *Abstract Routing Graph* (ARG) eingeführt, welcher alle durch *AnnounceCoordinator*-Nachrichten bekannten Koordinatorinstanzen für den Knoten speichert. Abbildung D.5 zeigt den ARG von Knoten 1:

1. Der L0-Cluster wird gespeichert, über den die entscheidende Nachricht empfangen wurde.
2. Die zum Knoten des entfernten Koordinators kürzeste Route wird gespeichert.
3. Für den entfernten Koordinator werden seine Identifikation sowie die Hop-Distanz zu seinem Knoten gespeichert.

Insbesondere für die Kontrolle des Ausbreitungsradius von *AnnounceCoordinator*-Nachrichten ist der ARG hilfreich.

D.3 Visualisierung der lokalen Entitäten und der Routingtabelle

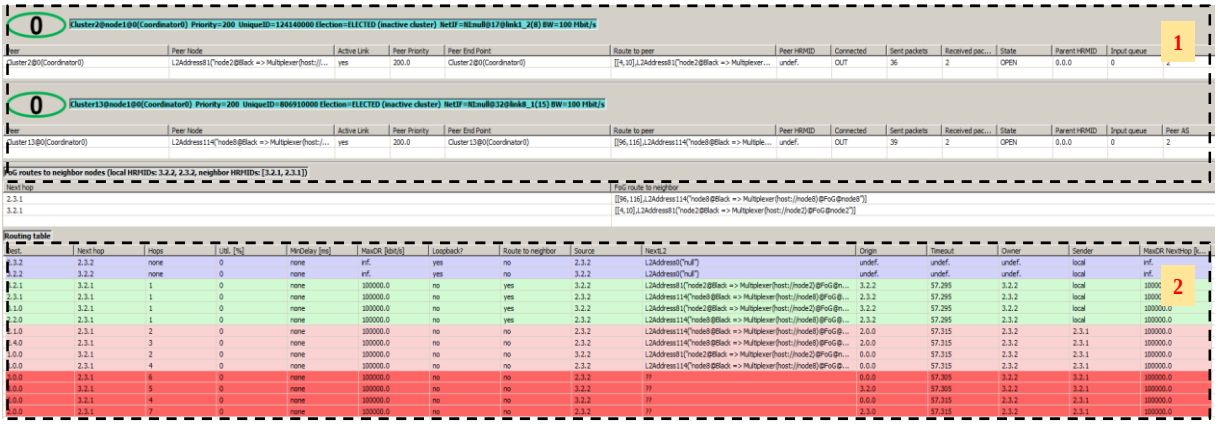

Abbildung D.6: Beispiel einer Übersicht über lokale Entitäten, FIB und Routingtabelle

Abbildung D.6 zeigt die lokalen Entitäten (1) sowie die Routingtabelle (2) von Knoten 1. Die Routingtabelle enthält innerhalb der HRM-Implementierung je Route die folgenden Werte:

- **Ziel:** die HRMID des Zielknotens/-clusters
- **Nächster Knoten:** die HRMID des nächsten Knotens der Route
- **Metrik: Hops:** die Hop-Distanz der Route
- **Metrik: Auslastung:** die prozentuale Gesamtauslastung der Route
- **Metrik: Verzögerung:** die zu erwartende minimale Verzögerung entlang der Route
- **Metrik Datenrate:** die maximal mögliche Datenrate entlang der Route
- **Markierung: „lokale Route“:** Ist es eine lokale Route?
- **Markierung: „Nachbarroute“:** Ist es eine Route zu einem direkten Nachbarn?
- **Quelle:** die lokale HRMID des Starts der Route
- **Nächste Knoten-ID:** die Knoten-ID des nächsten Knotens der Route
- **Ursprung:** die HRMID der Entität, welche die Daten der Route versendete
- **Timeout:** die absolute lokale Zeit, nach welcher die Route automatisch entfernt wird
- **Besitzer:** die HRMID der lokalen Entität, welche die Route ermittelte
- **Sender:** die HRMID des Senders der jeweiligen *RouteShare* Nachricht
- **Datenrate nächster Link:** die Datenrate des Links zum nächsten Knoten der Route

Durch diese im Vergleich zur Konzeption eher ausführlichen Routingtabelle wird die Nachvollziehbarkeit der Signalisierung zur Verteilung von Routingdaten unterstützt.

D.4 Darstellung eines Videostroms in FoGSiEm

Für das in *FoGSiEm* integrierte Videostreaming wird Homer zur Generierung des Videostroms benötigt, die Software überträgt die Videopakete mit Hilfe von UDP an *FoGSiEm*.

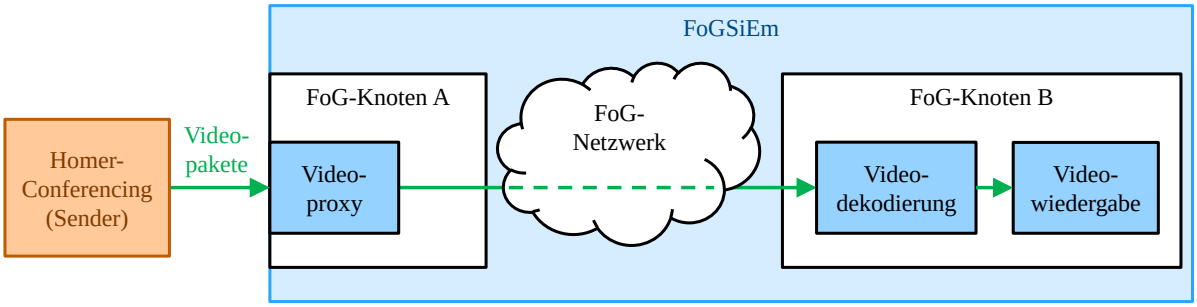

Abbildung D.7: Videostreaming in FoGSiEm

In Abbildung D.7 ist zu erkennen, dass innerhalb von *FoGSiEm* die UDP-Pakete durch eine Instanz eines Videoproxys auf FoG-Knoten A empfangen werden. Der abgebildete Videoproxy benutzt zu diesem Zweck einen lokal instanziierten Socket. Er ist für Pakete des physikalischen Netzwerks erreichbar. Alle eintreffenden Daten des Videostroms leitet der Videoproxy wiederum in unveränderter Form durch das simulierte Netzwerk weiter, dabei kann eine beliebige Routingimplementierung verwendet werden. Der Zielknoten der Paketweiterleitung im FoG-Netzwerk kann über grafische Dialoge in *FoGSiEm* ausgewählt werden. Angekommen auf dem Zielknoten B, wird der Videostrom mit Hilfe des Plug-Ins *fog.video* dekodiert. Anschließend wird er auf Basis des Plug-Ins *fog.video.view* durch ein Videowiedergabefenster auf dem Bildschirm angezeigt.

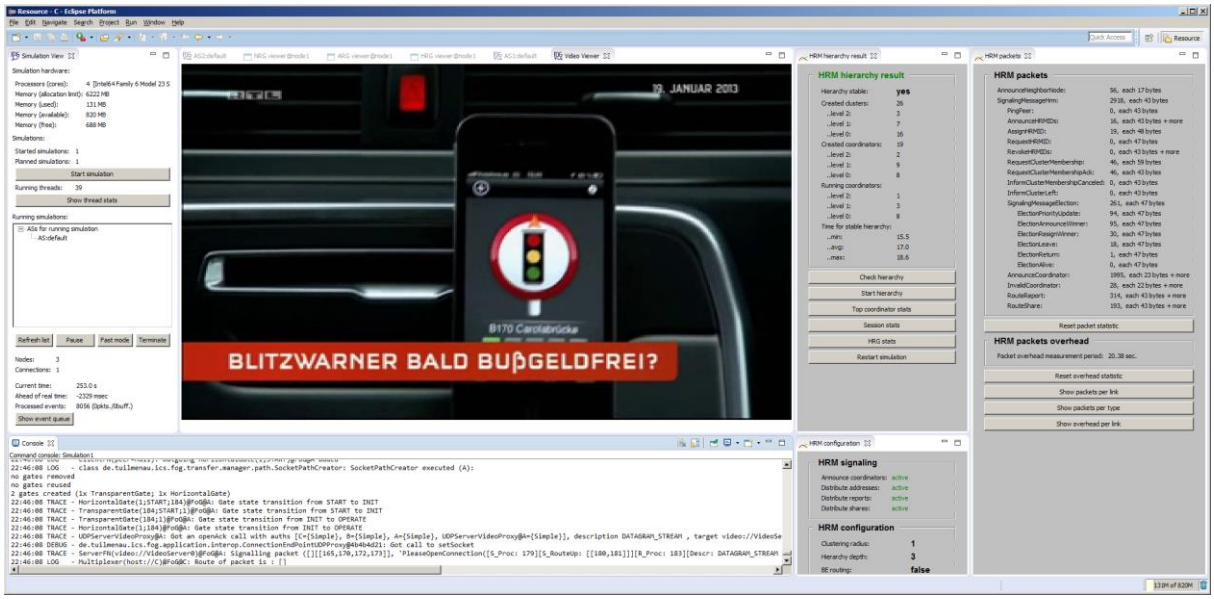

Abbildung D.8: Beispiel einer Videowiedergabe in FoGSiEm

Abbildung D.8 zeigt ein Videowiedergabefenster: Es ist das Bild des deutschen Fernsehsenders *RTL Television*[1] zu sehen. Über das Kontextmenü lassen sich zusätzliche Statistiken über den empfangenen Videostrom anzeigen, diese beinhalten neben einigen Codecparametern auch die gemessenen Paketverluste sowie die durchschnittliche Datenrate des Stroms. Diese Ausgaben dienen der Kontrolle der resultierenden Übertragungsqualität und lassen Rückschlüsse auf die gewählte Route zu, sodass qualitative Vergleiche zwischen verschiedenen Routingdiensten durchgeführt werden können.

[1] http://www.rtl.de

E Anwendung von Homer-Conferencing

Im Folgenden werden die grafischen Ausgaben von Homer anhand von Beispielen dargestellt. Dabei wird zum einen auf die explizite Erzeugung von Videoübertragungen eingegangen und zum anderen wird vorgestellt, wie ausgehende als auch eingehende audiovisuelle Datenströme einer Anwendungsinstanz überwacht werden können. Stellen Des Weiteren werden Details zur Bestimmung, automatischen Generierung und Übertragung von Qualitätsanforderungen gegeben. Der Anhang schließt mit einem ausgewählten Programmierbeispiel. Innerhalb des Anhangs wurde die Detailtiefe der Beschreibungen an den Fokus dieser Arbeit angepasst, für weitere Informationen wird auf die Webseite [11] verwiesen.

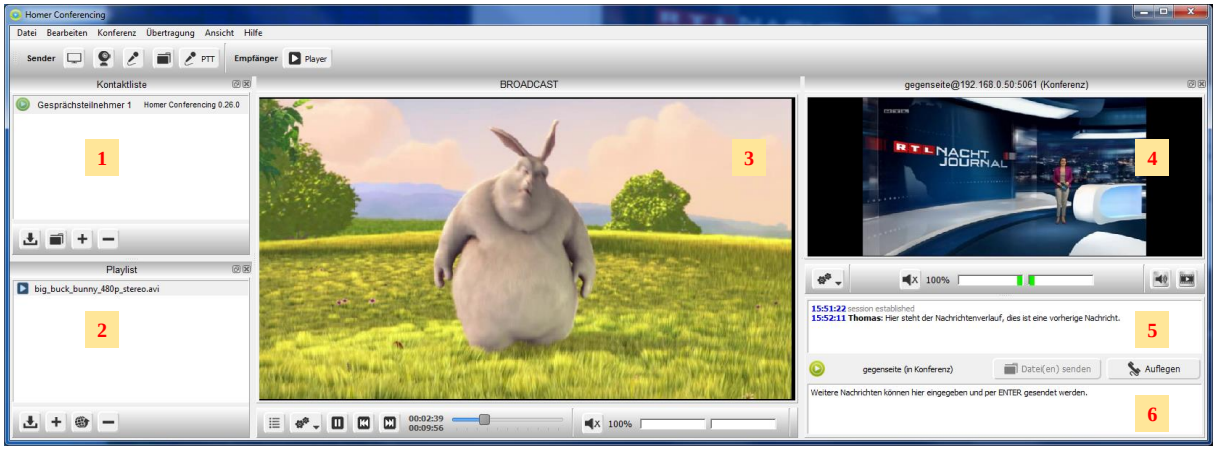

Abbildung E.1: Videokonferenzsitzung mit einem Gesprächspartner

In Abbildung E.1 ist eine Beispielsitzung zu sehen, sie zeigt ein Gespräch mit einem entfernten Konferenzteilnehmer. Die markierten Bereiche besitzen dabei folgende Bedeutung:

1. **Kontaktliste**: Hier werden alle bekannten Kontakte aufgeführt, diese können nach Bedarf per Nachricht oder Anruf kontaktiert werden.
2. **Playlist**: Über diese Liste können beispielsweise lokale audiovisuelle Dateien festgelegt werden, welche innerhalb der Konferenz übertragen und auf der jeweiligen Gegenseite wiedergegeben werden sollen.
3. **Broadcast**: Dieser Bereich gibt die lokale Video- und Audioquelle wieder, sodass eine Überwachung des ausgehenden Multimediainhaltes möglich ist.
4. **Video der Gegenseite**: Das von der Gegenseite empfangene Video wird in Echtzeit dargestellt, während der Audiostrom ebenfalls zu hören ist.
5. **Nachrichtenverlauf**: An dieser Stelle sind die bisher ausgetauschten Nachrichten zu sehen.
6. **Nachricht**: Die aktuelle Nachricht ist am unteren Rand zu sehen,

E.1 Grafische Dialoge zur Konfiguration audiovisueller Datenströme

Alle ausgehenden audiovisuellen Datenströme werden zentral über einen Konfigurationsdialog parametrisiert. Dabei können die jeweilige Quelle, der verwendete Codec sowie verschiedene Parameter für die Kodierung eingestellt werden.

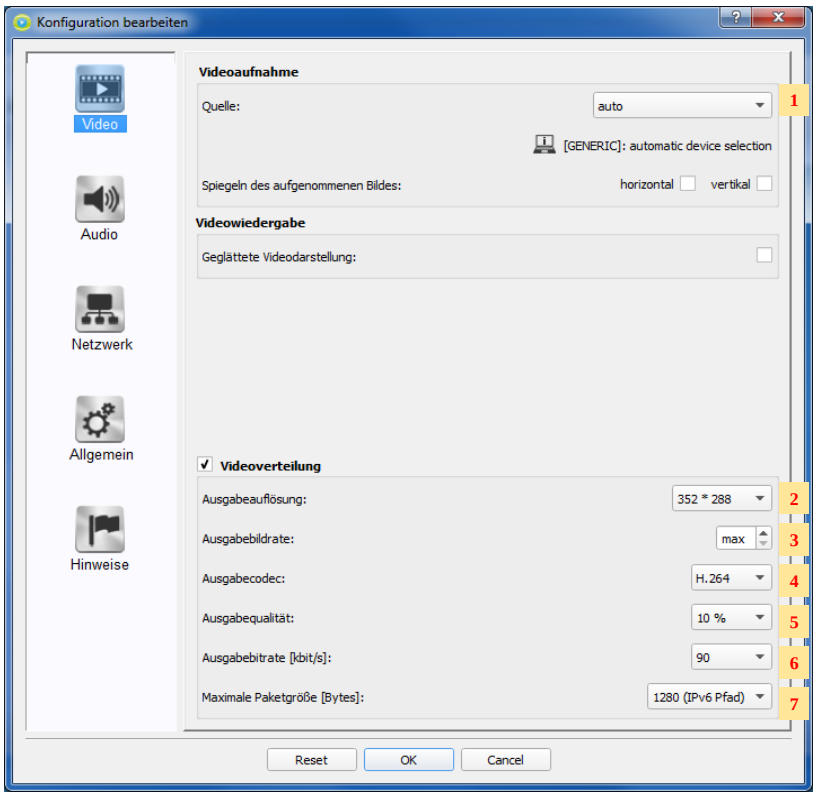

Abbildung E.2: Konfiguration der Videoparameter

Alle Parameter für die Videoverarbeitung sind in Abbildung E.2 zu sehen, dazu zählen:

1. **Quelle:** Das Eingangsvideo kann von einer lokalen Kamera, aus einer Datei oder vom lokalen Desktop abgegriffen werden.
2. **Ausgabeauflösung:** Es kann zwischen verschiedenen vorgegebenen Auflösungen gewählt werden, alternativ steht die Einstellung „auto“ zur Verfügung, welche automatisch die größte mögliche Auflösung auswählt.
3. **Ausgabebildrate:** Durch diese Einstellung kann die ausgehende Bildrate explizit reduziert werden.
4. **Ausgabecodec:** An dieser Stelle kann zwischen verschiedenen Codecs gewählt werden, dazu zählen H.2613/3+/4 und HEVC.
5. **Ausgabequalität:** Es sind Werte in Schritten von je 10% von 10% bis maximal 100% möglich.
6. **Ausgabebitrate:** Dieser Werte begrenzt die Datenrate für ausgehende Datenströme
7. **Maximale Paketgröße:** Durch diese Einstellung kann die maximale Größe der erzeugten Videopakete in Rahmen der technischen Möglichkeiten des jeweiligen Codecs festgelegt werden.

Ähnlich zu den in Abbildung E.2 dargestellten Möglichkeiten zur Parametrisierung der Videoverarbeitung kann ebenfalls die Audioverarbeitung konfiguriert werden, Details dazu sind der Homepage zu entnehmen.

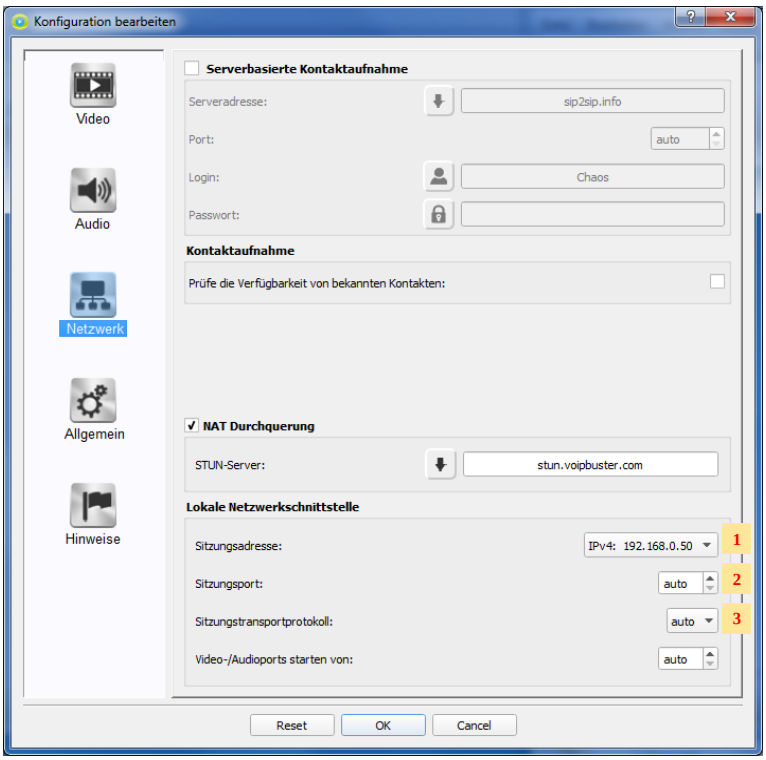

Abbildung E.3: Konfiguration der Netzwerkparameter

In Abbildung E.3 sind die Möglichkeiten zur Einstellung der wichtigsten Netzwerkparameter zu sehen, dazu zählen:

1. **Sitzungsadresse:** Die IPv4- oder IPv6-Adresse wird hier festgelegt, über welche die Anwendungsinstanz im Netzwerk erreichbar sein soll.
2. **Sitzungsport:** Der UDP-/TCP-Port für SIP-basierte Signalisierungen wird an dieser Stelle festgelegt.
3. **Sitzungsprotokoll:** Es kann zwischen UDP und TCP als Basis für die Verwaltung von aktiven Sitzungen gewählt werden.

Für weitere Details zu den nicht näher erläuterten Parametern sei wiederum auf die Homepage von Homer verwiesen.

E.2 Senden und Empfangen von Datenströmen

Mit Homer lassen sich explizit audiovisuelle Datenströme im Netzwerk erzeugen, dabei sind sowohl heutige IP-basierte Übertragungen als auch alternative Netzwerkstrukturen einsetzbar. Zu diesem Zweck wurde das Konzept der Programmierschnittstelle *G-Lab API* (GAPI) [136] innerhalb des Moduls *Network-API* (NAPI) umgesetzt, sodass die Datenübertragungen für beliebig viele Implementierungen eines Netzwerkstacks verwendet werden können. Zum Start einer Videoübertragung sind 4 Schritte notwendig:

1.) Start auf Senderseite
2.) Konfiguration auf Senderseite
3.) Konfiguration auf Empfängerseite
4.) Start der Verarbeitung auf Empfängerseite

Nachdem alle Schritte ausgeführt wurden, wird das empfangene Video beim Empfänger wiedergegeben. Die Schritte werden nachfolgend anhand der Dialoge detaillierter erläutert.

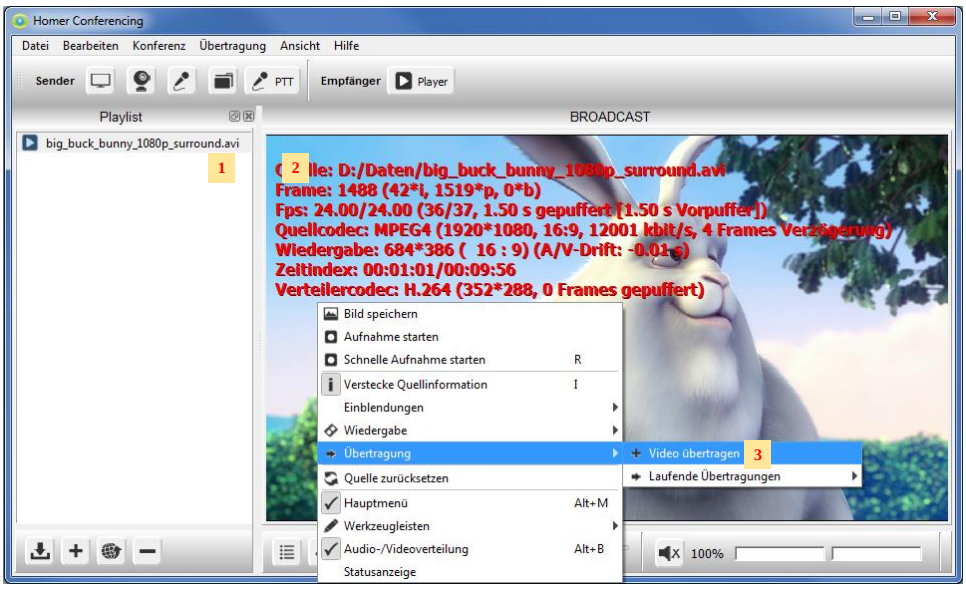

Abbildung E.4: Videostreaming – Start auf Senderseite

Die Abbildung E.4 zeigt den Ausgangszustand für Schritt 1:

1. **Playlist:** Das Video wurde für die Übertragung in die Liste geladen.
2. **Videowiedergabe:** Das aktuelle Video aus der Playlist wird sichtbar abgespielt, in roter Schrift werden optional verschiedene statistische Werte angezeigt.
3. **Videoübertragung:** Die Übertragung wird über das Kontextmenü gestartet.

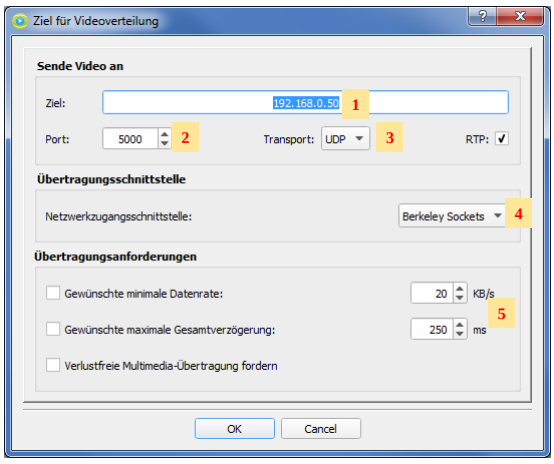

Abbildung E.5: Videostreaming – Konfiguration auf Senderseite

Schritt 2 wird in Abbildung E.5 dargestellt, es werden folgende Werte konfiguriert:

1. **Zieladresse:** Das kann beispielsweise die IP-Adresse oder der DNS-Name des Ziels sein.
2. **Port:** Der Zielport ist mit dem Standwert 5000 belegt und kann variiert werden, wobei diese Einstellung später auf Empfängerseite übernommen werden muss.
3. **Transport:** Als Transportprotokoll stehen UDP sowie TCP zur Verfügung. Insbesondere steht für Linux das Protokoll UDP-Lite als zusätzliche Option bereit. Das Protokoll SCTP wird insbesondere für den FoG-spezifischen Netzwerkstack verwendet.
4. **Netzwerkzugangsschnittstelle:** Mit Hilfe dieser Einstellung kann der gewünschte Netzwerkstack ausgewählt werden, die Standardeinstellung lautet „Berkeley-Sockets“.
5. **Übertragungsanforderungen:** Es kann explizit sowohl eine gewünschte Datenrate als auch eine maximal erlaubte Gesamtverzögerung gewählt werden.

Sobald der Dialog mit „OK“ bestätigt wird, startet die Videoübertragung durch das Netzwerk. Erst durch die Konfiguration und das Starten der Empfängerseite wird eine Wiedergabe am Ziel möglich, dabei können diese beiden Schritte bereits vor dem senderseitigen Start der Videoübertragung durchgeführt worden sein.

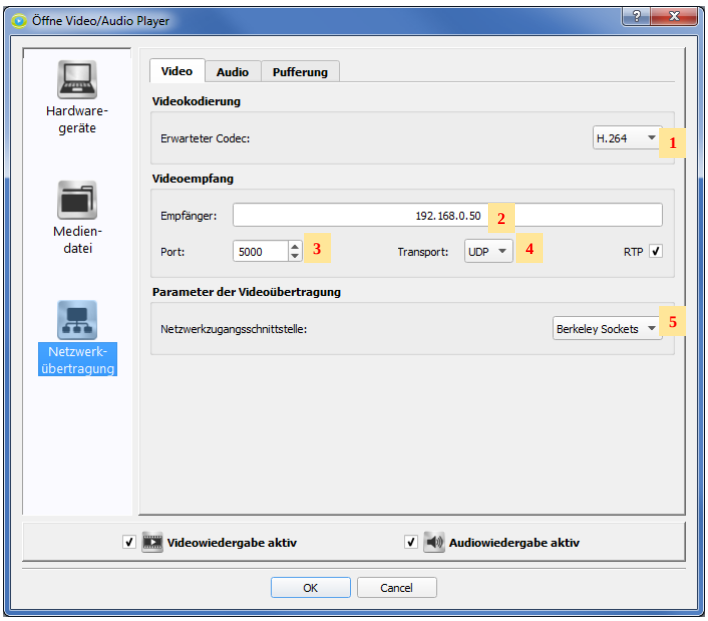

Abbildung E.6: Videostreaming – Konfiguration auf Empfängerseite

Die Konfiguration der Empfängerseite geschieht entsprechend Abbildung E.6 mit 5 Einstellungen:

1. **Videocodec:** Es muss der gleiche Codec wie auf der Senderseite eingestellt werden, um eine korrekte Wiedergabe zu ermöglichen. An dieser Stelle ist eine automatische Konfiguration des Videocodecs nicht möglich, da für diese Art der Videoübertragung keine Aushandlung von Mediaparametern, wie dies beispielsweise bei SIP üblich ist, angewandt wird.
2. **Empfängeradresse:** An dieser Stelle steht für eine IP-basierte Übertragung typischerweise eine IP-Adresse oder ein DNS-Name.
3. **Empfängerport:** Dieser Wert sollte dem senderseitig eingestellten Zielport entsprechen, andernfalls ist ein korrekter Empfang der Daten nicht möglich.
4. **Transport:** Als Transportprotokoll muss die senderseitige Einstellung verwendet werden. Sollte dabei TCP verwendet werden, muss die Senderseite nach dem Empfänger konfiguriert werden, andernfalls kann der Verbindungsaufbau nicht korrekt erfolgen.
5. **Netzwerkzugangsschnittstelle:** Über diese Einstellung wird der gewünschte Netzwerkstack ausgewählt, der Standartwert ist „Berkeley-Sockets“.

Sobald der Dialog mit „OK“ bestätigt wurde, startet der Videoempfang sowie die zugehörige Verarbeitung der eingehenden Daten, die Daten werden in Echtzeit wiedergegeben. Optional können zusätzliche Einstellungen für die Pufferung vorgenommen werden, sie werden im Folgenden erläutert.

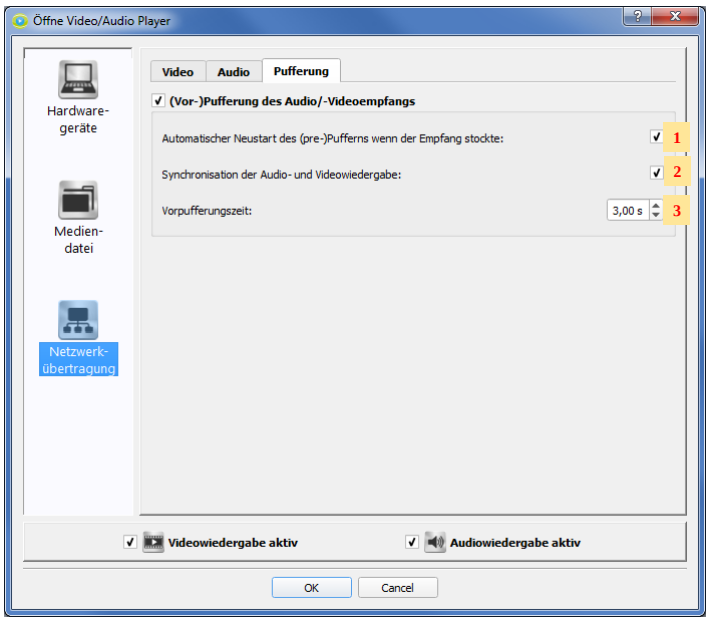

Abbildung E.7: Videostreaming – Konfiguration der Synchronisation auf Empfängerseite

Wie in Abbildung E.7 zu sehen, stehen 3 Optionen zur Verfügung:

1. Im Fall von sogenanntem *Stalling*, der Unterbrechung der Wiedergabe, kann die Pufferung automatisch neugestartet werden und die Wiedergabe wird zusätzlich verzögert fortgeführt.
2. Sollten neben einem Videostrom auch Audiodaten von der gleichen Quelle empfangen werden, kann die Wiedergabe beider Ströme durch Homer synchronisiert werden.
3. Die Länge des verwendeten Puffers für eintreffende Daten kann über eine Zeitangabe festgelegt werden, sie beschreibt die Wiedergabedauer der zu puffernden Daten.

Nachdem die Videoübertragung sowohl für den Sender als auch den Empfänger erfolgreich konfiguriert und gestartet wurde, erfolgt auf beiden Seiten die Wiedergabe in Echtzeit. Die Übertragung kann dabei beispielsweise durch ein von FoGSiEm simuliertes Netzwerk erfolgen.

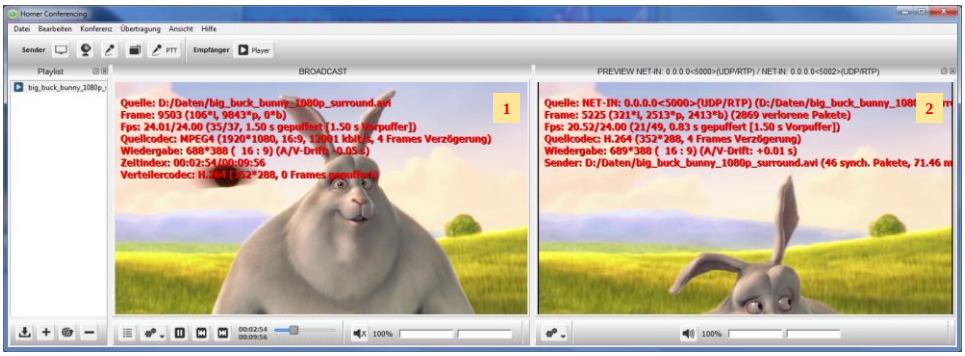

Abbildung E.8: Videostreaming – Sender und Empfänger im Vergleich

In Abbildung E.8 zeigt auf der linken Seite den Sender, während rechts daneben die Empfängerseite dargestellt wird. Somit ist ein direkter Vergleich zwischen Originalbild und übertragenen Bild möglich. In Abhängigkeit von der Wiedergabequalität auf Empfängerseite wird dadurch die Qualität der Übertragung deutlich, was wiederum Rückschlüsse auf den Erfolg des angewandten QoS-Routings ermöglicht.

E.3 Grafische Dialoge zur Überwachung von Datenströmen

Zur Überwachung ausgehender als auch eingehender audiovisueller Datenströme beinhaltet Homer eine entsprechend Sensorik innerhalb der Software, welche automatische Statistiken erstellt.

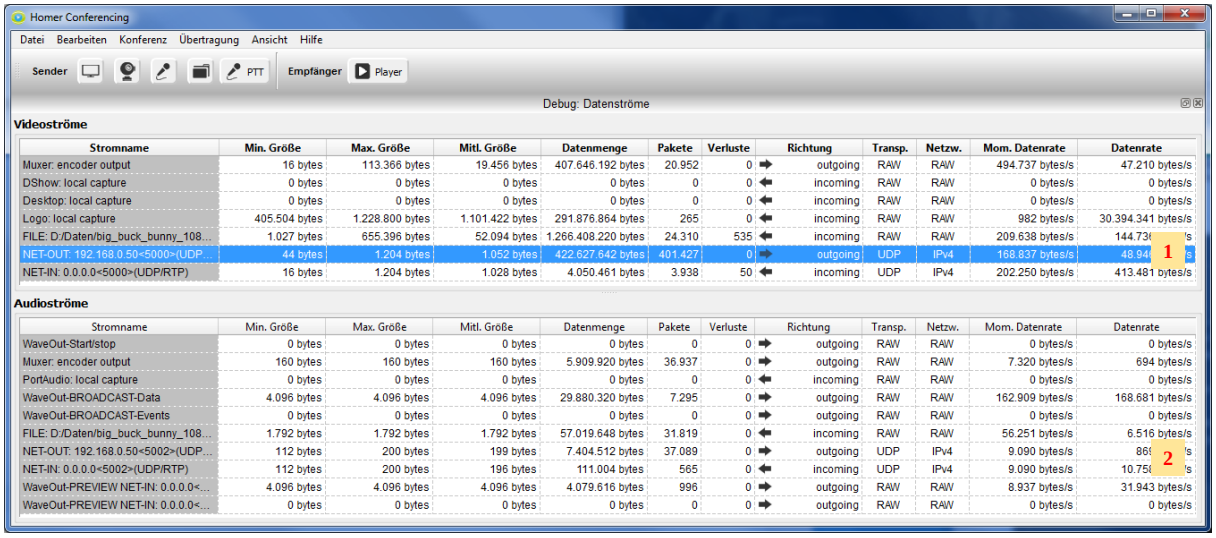

Abbildung E.9: Überwachung von allen audiovisuellen Datenströmen der Anwendungsinstanz

Wie in Abbildung E.9 ersichtlich können sowohl (1) Video- als auch (2) Audioströme beobachtet werden, dabei stehen verschiedene Messwerte von der Sensorik zur Verfügung:

1. minimale Paketgröße
2. maximale Paketgröße
3. mittlere Paketgröße
4. Datenmenge seit dem Start der Messung
5. Anzahl von erkannten Paketen
6. Anzahl von erkannten Paketverlusten (nur auf Empfängerseite)
7. Richtung der Übertragung (ausgehender oder eingehender Datenstrom)
8. Transport: Dabei kann es sich um einen sogenannten RAW-Datenstrom von/zu einem Gerät handeln. Des Weiteren können typische Transportprotokolle (bspw. UDP oder TCP) verwendet sein.
9. Netzwerk: Dabei kann es sich ebenfalls um einen sogenannten RAW-Datenstrom von/zu einem Gerät handeln. Alternativ kann IPv4 oder IPv6 zum Einsatz kommen.
10. momentane Datenrate (gemittelt über die letzten Pakete)
11. mittlere Datenrate für die gesamte Messung

Insbesondere die Anzeige der aufgetretenen Paketverluste ist wichtig, um Rückschlüsse auf die Übertragungsqualität ziehen zu können. Die dargestellten Werte beruhen dabei auf der Auswertung von eintreffenden RTP-Paketen.

E.4 Übertragung von Qualitätsanforderungen für den IP-Netzwerkstack

Dieser Abschnitt gibt zusätzliche Details über die in Abschnitt 5.5.1.2 erwähnte Lösung zur Übertragung von Qualitätsanforderungen innerhalb von IP-Netzwerken. Dabei werden für IPv4 zur Übertragung die sogenannten Optionen [13] verwendet, während für IPv6 stattdessen die sogenannten *Hop-by-Hop(HbH)*-Optionen [16] zum Einsatz kommen. Diese Erweiterungen sind als optionale Datenbereiche für die jeweilige IP-Version spezifiziert, durch ihre Verwendung erfolgt die Übertragung von Qualitätsanforderungen kompatibel zu existierenden IP-Implementierungen. Beide Varianten wurden in [159] detailliert untersucht, an dieser Stelle werden ausgewählte Teile des Ansatzes vorgestellt.

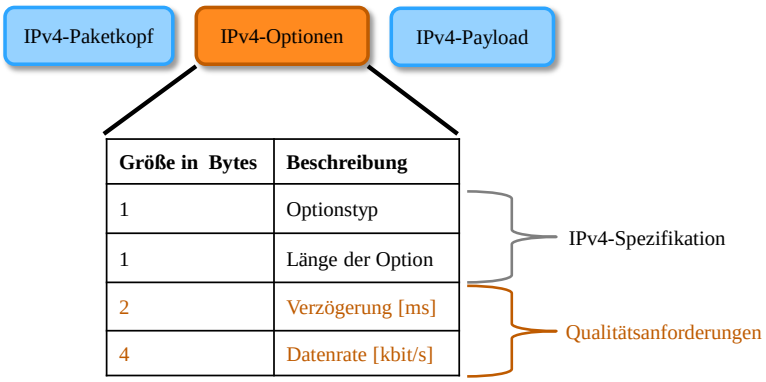

Abbildung E.10: *Inband*-Signalisierung von Qualitätsanforderungen bei IPv4

In Abbildung E.10 ist die Integration von Qualitätsanforderungen innerhalb von IPv4-Paketen zu sehen. Dabei werden 2 Bytes für die Kodierung der Verzögerung und 4 Bytes für die Datenrate verwendet, sodass dadurch akzeptable Wertebereiche für beide Werte zur Verfügung stehen.

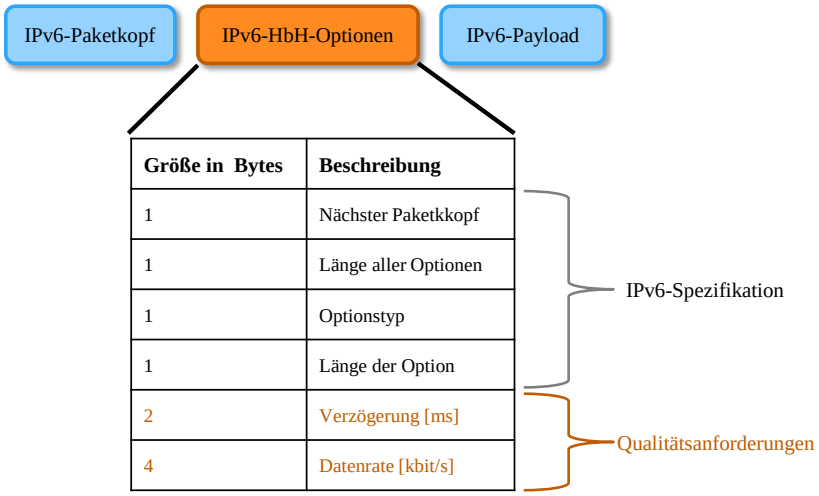

Abbildung E.11: *Inband*-Signalisierung von Qualitätsanforderungen bei IPv6

Abbildung E.11 zeigt die Integration in IPv6-Paketen, sie entspricht grundsätzlich der zuvor vorgestellten Variante für IPv4, beachtet jedoch die Besonderheiten der IPv6-Spezifikation. Wie zuvor werden 2 Bytes für die Verzögerung und 4 Bytes für die Datenrate verwendet.

In [159] wurden beide vorgestellten Varianten anhand einer Implementierung mit Hilfe von Homer-Conferencing untersucht. Dabei wurden auch die Grenzen des Ansatzes deutlich, nicht jede Implementierung eines Betriebssystems oder Routersoftware unterstützt die Weiterleitung der optionalen Daten innerhalb von IP-Paketen. Einige unterstützen nur ausgewählte Optionstypen, in Ausnahmefällen werden die zusätzlichen Daten gar nicht unterstützt. Als Folge daraus kann bei Verwendung neuer Optionstypen der Fall eintreten, dass Qualitätsanforderungen nicht an alle, oder an keinem, der passierten Netzwerkknoten übermittelt werden. Dies bedarf weiterer Untersuchungen, welche nicht Bestandteil dieser Arbeit sind.

E.5 Programmierbeispiel für eine Videoübertragung

Da Homer-Conferencing als Open-Source der Öffentlichkeit für weitere Experimente zur Verfügung steht, wird im Folgenden ein ausgewähltes Programmierbeispiel vorgestellt. Es beschreibt die notwendigen Schritte zur Übertragung eines Videostromes sowohl für die Sender- als auch die Empfängerseite. Auf Senderseite sind folgende Schritte notwendig:

1. **Initialisierung der benötigten Objekte:**

   *MediaSourceMuxer *mux = new MediaSourceMuxer();*
   *MediaSourceV4L2 *v4l2 = new MediaSourceV4L2();*
   mux ->RegisterMediaSource(v4l2);
   mux ->SetOutputStream("H.264", ...);

   Durch die Anweisungen wird die Kameraquelle *v4l2* erzeugt (sie benutzt die Linux-spezifische Schnittstelle *Video4Linux2*), welche bei der erzeugten Instanz *mux* eines Muxers registriert wird, sodass dadurch der erste Teil der Verarbeitungskette bereits gültig ist. Zusätzlich wird für den Muxer explizit der Videocodec H.264 gewählt.

2. **Start des Muxers:**

   mux ->OpenVideoGrabDevice(...);

   Dadurch wird ebenfalls der Zugriff auf die registrierte Kameraquelle gestartet.

3. **Registrierung einer Netzwerksenke:**

   mux ->RegisterMediaSink(„::1", 5000, RTP_UDP);

   Eine Netzwerksenke wird am Muxer registriert, sodass jedes nachfolgende Videoframe, welches den Muxer passiert, in RTP-Pakete aufgeteilt und an den UDP-Port 5000 gesendet wird.

4. **Abfrage des nächsten Videoframes:**

   mux ->GrabChunk(...);

   Diese Anweisung muss kontinuierlich in einem eigenständigen Thread wiederholt werden, sodass jeweils ein neues Videoframe von der Kameraquelle an den Muxer geschickt wird. Da bereits in Schritt 3 eine Netzwerksenke angelegt und registriert wurde, wird das Frame durch die Verarbeitung im Muxer ebenfalls an den Empfänger im Netzwerk übertragen.

Die empfangende Gegenseite kann bei Verwendung von UDP entweder vor dem Sender oder danach gestartet werden. Die Empfängerseite muss die folgenden Schritte ausführen:

1. **Initialisierung der benötigten Objekte:**

   *MediaSourceNet *net = new MediaSourceNet("::", 5000, RTP_UDP);*
   net ->SetInputStream("H.264", ...);

   Durch die Anweisungen wird ein Objekt vom Typ MediaSourceNet erzeugt, welches die Netzwerk-basierte Videoquelle darstellt. Sie nimmt RTP-Pakete an UDP-Port 5000 entgegen und generiert automatisch Videoframes. Für diese Verarbeitung wurde explizit der Videocodec H.264 festgelegt.

2. **Abfrage des nächsten Videoframes:**

   net ->GrabChunk(...);

   Diese Anweisung muss kontinuierlich in einem eigenständigen Thread wiederholt werden, sodass jeweils ein neues Videoframe auf Basis des über das Netzwerk empfangenen Datenstroms ermittelt wird. Es kann nachfolgend beispielsweise in einem Qt-basierten Videofenster angezeigt werden.

Ähnlich des beschriebenen Vorgehens kann für die Übertragung von Audioströmen vorgegangen werden, sodass dadurch auf sehr einfache Art ein qualitativer Vergleich für unterschiedliche Routingalgorithmen durchgeführt werden kann. Sowohl die Bild- als auch die Tonwiedergabe auf Empfängerseite lassen dabei Rückschlüsse auf die Qualität der verwendeten Route und der daraus resultierenden Übertragung zu.

F Simulationshardware

Hardware	A	B	C	D	E
Standort	Rechenzentrum	Rechenzentrum	Rechenzentrum	privat	Büro
Prozessor	Intel XEON X5660	Intel XEON E5-2609 v2	AMD Opteron 23xx	Intel Core 2 Quad Q9550	Intel Core i7 960
Prozessortakt	2,8 GHz	2,5 GHz	2,2 GHz	2,83 GHz	3,2 GHz
Prozessorkerne	12	8	20	4	8
Speicher	24 GB	64	90 GB	8 GB	6 GB
Virtuelles System	nein	nein	ja	nein	nein

Tabelle F.1: Eigenschaften der Simulationshardware

Messungstyp	A	B	C	D	E
Startphase der Kontrollebene	X				
Betriebsphase der Kontrollebene	X	X	X	X	X
QoS-Routing der Datenebene	X	X		X	

Tabelle F.2: Verwendung der Simulationshardware

Als Simulationshardware dienten 5 unterschiedliche PCs und Server, welche jeweils eine unterschiedliche Konfiguration besaßen. Tabelle F.1 gibt einen Überblick über die Eigenschaften der Hardware. In Abhängigkeit von der Leistungsfähigkeit des jeweiligen Prozessors wurde sie für unterschiedliche Messungstypen verwendet, Tabelle F.2 zeigt die Zuordnung. Für möglichst gute Vergleichswerte der für die Startphase notwendigen Signalisierungen wurde für diese Experimente ausschließlich Hardware A verwendet, bei den sonstigen Messtypen war eine derartige Überlegung nicht notwendig.

Abbildungsverzeichnis

Tabellenverzeichnis

Formelverzeichnis

Index

Literaturverzeichnis

[1] Cisco Systems, „Visual Networking Index: Forecast and Methodology, 2011–2016," 2012.

[2] Cisco Systems, „Visual Networking Index: Forecast and Methodology, 2013–2018," 2014.

[3] Sandvine Incorporated ULC, „Global Internet Phenomena Report: 1H 2014," 2014. [Online]. Available: https://www.sandvine.com/downloads/general/global-internet-phenomena/2014/1h-2014-global-internet-phenomena-report.pdf. [Zugriff am 23. März 2015].

[4] K. Nichols, S. Blake, F. Baker und D. Black, „Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," IETF RFC 2474, 1998.

[5] J. Wroclawski, „The Use of RSVP with IETF Integrated Services," IETF RFC 2210, 1997.

[6] R. Coltun, D. Ferguson, J. Moy und A. Lindem, „OSPF for IPv6," IETF RFC 5340, 2008.

[7] Zhang, Sanchez, Salkewicz und Crawley, „Quality of Service Extensions to OSPF or Quality of Service Path First Routing (QOSPF)," IETF Draft, 1997.

[8] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda und T. Przygienda, „QoS Routing Mechanisms and OSPF Extensions," IETF RFC 2676, 1999.

[9] F. Liers, T. Volkert und A. Mitschele-Thiel, „The forwarding on gates architecture: Merging intserv and diffserv," *4th International Conference on Advances in Future Internet (AFIN)*, pp. 7-13, 2012.

[10] „FoGSiEm - Webpräsenz auf GitHub," 2013. [Online]. Available: https://github.com/ICS-TU-Ilmenau/fog/wiki. [Zugriff am 23. März 2015].

[11] T. Volkert, „Homer - live conferencing and more," 2015. [Online]. Available: http://www.homer-conferencing.com. [Zugriff am 23. März 2015].

[12] J. Day, „Patterns in Network Architecture - A Return to Fundamentals," Prentice Hall, 2008.

[13] J. Postel, „Internet Protocol - DARPA Internet Program Protocol Specification," IETF RFC 791, 1981.

[14] P. Mockapetris, „Damain Names - Concepts and Facilities," IETF RFC 1034, 1987.

[15] „Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model," ISO/IEC 7498-1, 1994.

[16] S. Deering und R. Hinden, „Internet Protocol, Version 6 (IPv6) Specification," IETF RFC 2460, 1998.

[17] Internet Assigned Numbers Authority (IANA), „Service Name and Transport Protocol Port Number Registry," 2015. [Online]. Available: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml. [Zugriff am 23. März 2015].

[18] J. Postel, „Transmission Control Protocol - Darpa Internet Program - Protocol Specification," IETF RFC 793, 1981.

[19] J. Postel, „User Datagram Protocol," IETF RFC 768, 1980.

[20] R. Stewart, „Stream Control Transmission Protocol," IETF RFC 4960, 2007.

[21] G. Lienemann, „TCP/IP - Grundlagen. Protokolle und Routing," Verlag Heinz Heise, 2000.

[22] IEEE 802.3 Working Group, „802.3-2012 - IEEE Standard for Ethernet," 2012. [Online]. Available: http://standards.ieee.org/about/get/802/802.3.html. [Zugriff am 23. März 2015].

[23] Cisco Systems, „Cisco Networking Academy Program - CCNA 3: Grundlagen des Switching und Intermediate Routing," Kursunterlagen zum Erwerb der CCNA-Zertifizierung, 2003.

[24] D. Medhi und K. Ramasamy, „Network Routing - Algorithm, Protocols, and Architectures," Morgan Kaufmann Publishers, 2007.

[25] R. Hinden und S. Deering, „Internet Protocol Version 6 (IPv6) Addressing Architecture," IETF RFC 3513, 2003.

[26] M. Cotton, L. Vegoda, R. Bonica und B. Haberman, „Special-Purpose IP Address Registries," IETF RFC 6890, 2013.

[27] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot und E. Lear, „Address Allocation for Private Internets," IETF RFC 1918, 1996.

[28] R. Hinden und B. Haberman, „Unique Local IPv6 Unicast Addresses," IETF RFC 4193, 2005.

[29] B. Croft und J. Gilmore, „Bootstrap Protocol (BOOTP)," IETF RFC 951, 1985.

[30] R. Droms, „Dynamic Host Configuration Protocol," IETF RFC 2131, 1997.

[31] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins und M. Carney, „Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," IETF RFC 3315, 2003.

[32] W. Simpson, „The Point-to-Point Protocol (PPP)," IETF RFC 1661, 1994.

[33] L. Mamakos, K. Lidl, J. Evarts, D. Carrel, D. Simone und R. Wheeler, „A Method for Transmitting PPP Over Ethernet (PPPoE)," IETF RFC 2516, 1999.

[34] S. Thomson, T. Narten und T. Jinmei, „IPv6 Stateless Address Autoconfiguration," IETF RFC 4862, 2007.

[35] T. Narten, E. Nordmark, W. Simpson und H. Soliman, „Neighbor Discovery for IP version 6 (IPv6)," IETF RFC 4861, 2007.

[36] J. Moy, „OSPF Version 2,“ IETF RFC 2328, 1998.

[37] D. C. Plummer, „An Ethernet Address Resolution Protocol - or - Converting Network Protocol Addresses,“ IETF RFC 826, 1982.

[38] E. C. Rosen, „Exterior Gateway Protocol (EGP),“ IETF RFC 827, 1982.

[39] J. Hawkinson und T. Bates, „Guidelines for creation, selection, and registration of an Autonomous System (AS),“ IETF RFC 1930, 1996.

[40] S. Hares und D. Katz, „Administrative Domains and Routing Domains - A Model for Routing in the Internet,“ IETF RFC 1136, 1989.

[41] G. Malkin, „RIP Version 2,“ IETF RFC 2453, 1998.

[42] Y. Rekhter, T. Li und S. Hares, „A Boarder Gateway Protocol 4 (BGP-4),“ IETF RFC 4271, 2006.

[43] M. Caesar und J. Rexford, „BGP routing policies in ISP networks,“ *IEEE Network,* Bd. 19, Nr. 6, pp. 5-11, November 2005.

[44] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu und L. Zhang, „IPv4 address allocation and the BGP routing table evolution,“ *ACM SIGCOMM Computer Communication Review,* Bd. 35, Nr. 1, pp. 71-80, Januar 2005.

[45] T. Bates, E. Chen und R. Chandra, „BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP),“ IETF RFC 4456, 2006.

[46] P. Traina, D. McPherson und J. Scudder, „Autonomous System Confederations for BGP,“ IETF RFC 5065, 2007.

[47] K.-O. Detken, „Echtzeitplattformen für das Internet - Grundlagen, Lösungsansätze der sicheren Kommunikation mit QoS und VoIP,“ Addison-Wesley, 2002.

[48] R. Braden, L. Zhang, S. Berson, S. Herzog und S. Jamin, „Resource ReSerVation Protocol (RSVP),“ IETF RFC 2205, 1997.

[49] X. Fu, H. Schulzrinne, A. Bader, D. Hogrefe, C. Kappler, G. Karagiannis, H. Tschofeng und S. Van den Bosch, „NSIS: a new extensible IP signaling protocol suite,“ *IEEE Communications Magazine,* Bd. 43, Nr. 10, pp. 133-141, Oktober 2005.

[50] W. Zhao, D. Olshefski und H. Schulzrinne, „Internet Quality of Service: An Overview,“ *Columbia University Computer Science Technical Reports,* 2000.

[51] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski und E. Felstaine, „A Framework for Integrated Services Operation over Diffserv Networks,“ IETF RFC 2998, 2000.

[52] Z. Mammeri, „End-to-End QoS Mapping in IntServ-over-DiffServ Architectures," *6th IEEE International Conference on High-Speed Networks and Multimedia Communications,* pp. 31-40, 2003.

[53] Z. Mammeri, „End-to-end QoS mapping in IntServ-over-DiffServ architectures," *High-Speed Networks and Multimedia Communications,* Bd. 2720, pp. 31-40, 2003.

[54] P. Chandra, A. Fisher, C. Kosak, T. S. Ng, P. Steenkiste, E. Takahashi und H. Zhang, „Darwin: customizable resource management for value-added network services," *6th International Conference on Network Protocols,* pp. 177-188, Oktober 1998.

[55] „Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 2: Video," ISO/IEC 11172-2:1993, 1993.

[56] T. Hoßfeld, M. Fiedler und T. Zinner, „The QoE provisioning-delivery-hysteresis and its importance for service provisioning in the Future Internet," *7th EURO-NGI Conference on Next Generation Internet (NGI),* pp. 1-8, Juni 2011.

[57] K. Yamagishi und T. Hayashi, „Parametric Packet-Layer Model for Monitoring Video Quality of IPTV Services," *IEEE International Conference on Communications (ICC),* pp. 110-114, Mai 2008.

[58] T. Hoßfeld, F. Liers, R. Schatz, B. Staehle, D. Staehle, T. Volkert und F. Wamser, „Quality of Experience Management for YouTube: Clouds, FoG and the AquareYoum," *PIK - Praxis der Informationsverarbeitung und Kommunikation,* Bd. 35, Nr. 3, pp. 133-143, 2012.

[59] International Telecommunications Union, „G.114: One-way transmission time," ITU-T G.114, 2003.

[60] F. Geogatos, F. Gruber, D. Karrenberg, M. Santcroos, A. Susanj, H. Uijterwaal und R. Wilhelm, „Providing Active Measurements as a Regular Service for ISP's," *Passive & Active Measurement (PAM),* pp. 45-56, 2001.

[61] C. J. Bovy, H. T. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal und P. Van Mieghem, „Analysis of End-to-end Delay Measurements in Internet," *Passive and Active Measurement (PAM),* 2002.

[62] G. R. Ash, Traffic Engineering and QoS Optimization of Integrated Voice & Data Networks, Morgan Kaufmann, 2007.

[63] Z. Wang und J. Crowcroft, „Quality-of-service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communication,* Bd. 14, Nr. 7, pp. 1228-1234, 1996.

[64] S. Uludag, K.-S. Lui, K. Nahrstedt und G. Brewster, „Analysis of Topology Aggregation techniques for QoS routing," *ACM Computing Surveys (CSUR),* Bd. 39, Nr. 3, 2007.

[65] D. Katz, K. Kompella und D. Yeung, „Traffic Engineering (TE) Extensions to OSPF Version 2," IETF RFC 3630, 2003.

[66] ATM Forum, „Private Network-Network Interface," Version 1.0, 1996.

[67] D. E. McDysan und D. Paw, ATM & MPLS Theory & Application: Foundations of Multi-Service Networking, Osborne/McGraw-Hill, 2002.

[68] W. C. Lee, „Topology aggregation for hierarchical routing in ATM networks," *ACM SIGCOMM Computer Communication Reviews,* Bd. 25, Nr. 2, pp. 82-92, 1995.

[69] A. Farrel, A. Satyanarayana, A. Iwata, N. Fujita und G. Ash, „Crankback Signaling Extensions for MPLS and GMPLS RSVP-TE," IETF RFC 4920, 2007.

[70] O. Younis und S. Fahmy, „Constraint-Based Routing in the Internet: Basic Principles and Recent Research," *IEEE Communications Surveys & Tutorials,* Bd. 5, Nr. 1, pp. 2-13, 2003.

[71] L. Hanzo II und R. Tafazolli, „A survey of QoS routing solutions for mobile ad hoc networks," *IEEE Communications Survey & Tutorials,* Bd. 9, Nr. 2, pp. 50-70, 2007.

[72] L. Chen und W. B. Heinzelman, „A Survey of Routing Protocols that Support QoS in Mobile Ad Hoc Networks," *IEEE Network,* Bd. 21, Nr. 6, pp. 30-38, 2007.

[73] X. Zou, B. Ramamurthy und S. Magliveras, „Routing Techniques in Wireless Ad Hoc Networks - Classification and Comparison," *6th World Multiconference on Systemics, Cybernetics, and Informatics (SCI),* 2002.

[74] S. Sundar, R. Kumar, H. M. Kittur und M. Shanmugasundaram, „Manet Routing Protocols with QoS Support-A Survey," *International Journal of Engineering & Technology,* Bd. 5, Nr. 3, 2013.

[75] G. Santhi und A. Nachiappan, „A Survey of QoS Routing Protocols for Mobile Ad Hoc Networks," *International journal of computer science & information Technology (IJCSIT),* Bd. 2, Nr. 4, 2010.

[76] P. Becker, „QoS Routing Protocols for Mobile Ad-hoc Networks - A Survey," Technischer Report, TU Kaiserslautern, 2007.

[77] K. Akkaya und M. Younis, „A Survey on Routing Protocols for Wireless Sensor Networks," *Ad hoc networks (Elsevier),* Bd. 3, Nr. 3, pp. 325-349, 2005.

[78] G. M. Shafiullah, A. Gyasi-Agyei und P. J. Wolf, „A Survey of Energy-Efficient and QoS-Aware Routing Protocols for Wireless Sensor Networks," *Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics,* pp. 352-357, 2008.

[79] R. Sumathi und M. G. Srinivas, „A Survey of QoS Based Routing Protocols for Wireless Sensor Networks," *Journal of Information Processing Systems,* Bd. 8, Nr. 4, pp. 589-602, 2012.

[80] F. A. Kuipers, P. Van Mieghem, T. Korkmaz und M. Krunz, „An overview of constraint-based path selection algorithms for QoS routing," *IEEE Communications Magazine,* Bd. 40, Nr. 12, pp. 50-55, 2002.

[81] P. Van Mieghem und F. A. Kuipers, „Concepts of exact QoS routing algorithms," *IEEE/ACM Transactions on Networking,* Bd. 12, Nr. 5, pp. 851-864, 2004.

[82] M. K. Gulati und K. Kumar, „Survey of Multipath QoS Routing Protocols for Mobile Ad Hoc Networks," *International Journal of Advances in Engineering & Technology,* Bd. 3, Nr. 2, pp. 809-819, 2012.

[83] S. Chen und K. Nahrstedt, „An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions," *IEEE Network,* Bd. 12, Nr. 6, pp. 64-79, 1998.

[84] S. Chen und K. Nahrstedt, „Distributed quality-of-service routing in high-speed networks based on selective probing," *23rd Annual Conference on Local Computer Networks,* pp. 80-89, 1998.

[85] S. Chen und K. Nahrstedt, „Distributed QoS routing with imprecise state information," *7th International Conference on Computer Communications and Networks,* pp. 614-621, 1998.

[86] A. Shaikh, J. Rexford und K. G. Shin, „Dynamics of quality-of-service routing with inaccurate link-state information," Technischer Report CSE-TR-350-97, Universität Michigan, 1997.

[87] D. Krioukov, K. Fall und X. Yang, „Compact routing on Internet-like graphs," *23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM),* 2004.

[88] L. Kleinrock und F. Kamoun, „Hierarchical routing for large networks performance evaluation and optimization," *Computer Networks (Elsevier),* Bd. 1, Nr. 3, pp. 155-174, 1976.

[89] J. M. McQuillan, „Adaptive routing algorithms for distributed computer networks," *DTIC Dokument,* 1974.

[90] Z. J. Haas, „A new routing protocol for the reconfigurable wireless networks," *6th International Conference on Universal Personal Communications,* Bd. 2, pp. 562-566, 1997.

[91] G. Pei, M. Gerla, X. Hong und C.-C. Chian, „A wireless hierarchical routing protocol with group mobility," *Wireless Communications and Networking Conference,* p. 15381542, 1999.

[92] F. Liers, T. Volkert und A. Mitschele-Thiel, „The Forwarding on Gates Architecture: Flexible Placement of QoS Functions and States in Inter-Networks," *International Journal On Advances in Internet Technology,* Bd. 6, Nr. 3, pp. 132-144, 2013.

[93] F. Liers, Forwarding on Gates - A flexible and scalable inter-network layer supporting in-network functions (Dissertation), Universitätsverlag Ilmenau, 2014.

[94] L. Volkert, D. Martin, I. El Khayaut, C. Werle und M. Zitterbart, „A node architecture for 1000 future networks," *International Conference on Communications (ICC) - Workshops,* pp. 1-5, 2009.

[95] P. Müller und B. Reuther, „Future Internet Architecture--A Service Oriented ApproachFuture Internet Architecture--Ein serviceorientierter Ansatz," *it-Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik,* Bd. 50, Nr. 6, pp. 383-389, 2009.

[96] R. Khondoker, A. Siddiqui, B. Reuther und P. Müller, „Service Orientation Paradigm in Future Network Architectures,“ *6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS),* pp. 346-351, 2012.

[97] P. Godfrey, I. Ganichev, S. Shenker und I. Stoica, „Pathlet routing,“ *ACM SIGCOMM Computer Communication Review,* Bd. 39, Nr. 4, pp. 111-122, 2009.

[98] S. Kögel, „Hierarchisches Routing im Kontext des Future Internet Ansatzes "Forwarding on Gates" (Diplomarbeit),“ TU Ilmenau, 2010.

[99] M. Osdoba, „Evaluierung eines hierarchischen Routingsystems im Kontext des Future Internet Ansatzes "Forwarding on Gates" (Masterarbeit),“ TU Ilmenau, 2012.

[100] T. Volkert, M. Osdoba, M. Becke und A. Mitschele-Thiel, „Multipath Video Streaming Based on Hierarchical Routing Management,“ *3rd International Workshop on Protocols and Applications with Multi-Homing Support (PAMS) - 27th International Conference on Advanced Information Networking and Applications (AINA),* pp. 1-6, 2013.

[101] J. Yu, „Scalable Routing Design Principles,“ IETF RFC 2791, 2000.

[102] P. Leach, M. Mealling und R. Salz, „A Universally Unique IDentifier (UUID) URN Namespace,“ IETF RFC 4122, 2005.

[103] H. Garcia-Molina, „Elections in a distributed computing system,“ *IEEE Transactions on Computers,* Bd. 100, Nr. 1, pp. 48-59, 1982.

[104] R. Xu und D. Wunsch, „Survey of clustering algorithms,“ *IEEE Transactions on Neural Networks,* Bd. 16, Nr. 3, pp. 645-678, 2005.

[105] D. Mills, J. Martin, J. Burbank und W. Kasch, „Network Time Protocol Version 4: Protocol and Algorithms Specifications,“ IETF RFC 5905, 2010.

[106] F. Bian, X. Li, R. Govindan und S. Shenker, „Using Hierarchical Location Names for Scalable Routing and Rendezvous in Wireless Sensor Networks,“ *International Journal of Ad Hoc and Ubiquitous Computing,* Bd. 1, Nr. 4, pp. 179-193, 2006.

[107] J. Nagle, „Congestion Control in IP/TCP Internetworks,“ IETF RFC 896, 1984.

[108] J. Postel, „Transmission Control Protocol,“ IETF RFC 793, 1981.

[109] C. Villamizar, R. Chandra und R. Govindan, „BGP Route Flap Damping,“ IETF RFC 2439, 1998.

[110] L. Subramanian, I. Stoica, H. Balakrishnan und R. H. Katz, „Overqos: offering internet qos using overlays,“ *ACM SIGCOMM Computer Communication Review,* Bd. 33, Nr. 1, pp. 11-16, 2003.

[111] R. Bless, C. Mayer, C. Hübsch und O. Waldhorst, „Spovnet: an architecture for easy creation and deployment of service overlays,“ *River Publishers,* Bd. 6, pp. 23-47, 2011.

[112] O. Waldhorst, C. Blankenhorn, D. Haage, R. Holz, G. G. Koch, B. Koldehofe, F. Lampi, C. Mayer und S. Mies, „Spontaneous Virtual Networks: On the Road Towards the Internet's Next Generation," *it--Information Technology,* Bd. 50, pp. 367-375, 2008.

[113] T. Li und Y. Rekhter, „A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)," IETF RFC 2430, 1998.

[114] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan und G. Swallow, „RSVP-TE: Extensions to RSVP for LSP Tunnels," IETF RFC 3209, 2001.

[115] A. Guénoche, P. Hansen und B. Jaumard, „Efficient algorithms for divisive hierarchical clustering with the diameter criterion," *Journal of Classification,* Bd. 8, Nr. 1, pp. 5-30, 1991.

[116] A. Das und C. Kenyon-Mathieu, „On Hierarchical Diameter-Clustering and the Supplier Problem," *Theory of Computing Systems,* Bd. 45, Nr. 3, pp. 497-511, 2009.

[117] S. A. Awwad, C. K. Ng, N. K. Noordin und M. F. A. Rasid, „Cluster based routing protocol for mobile nodes in wireless sensor network," *Wireless Personal Communications,* Bd. 61, Nr. 2, pp. 251-281, 2011.

[118] W. Heinzelman, A. Chandrakasan und H. Balakrishnan, „Energy-efficient communication protocol for wireless microsensor networks," *33rd annual Hawaii international conference on system sciences,* 2000.

[119] L. Arboleda und N. Nasser, „Cluster-based routing protocol for mobile sensor networks," *3rd international conference on quality of service in heterogeneous wired/wireless networks,* 2006.

[120] Y. Yin, J. Shi, Y. Li und P. Zhang, „Cluster head selection using analytical hierarchy process for wireless sensor networks," *17th International Symposium on Personal, Indoor and Mobile Radio Communications,* pp. 1-5, 2006.

[121] F. Bian, X. Li, R. Govindan und S. Shenker, „Using hierarchical location names for scalable routing and rendezvous in wireless sensor networks," *International Journal of Ad Hoc and Ubiquitous Computing,* Bd. 1, Nr. 4, pp. 179-193, 2006.

[122] D. Johnson und D. Maltz, „Dynamic source routing in ad hoc wireless networks," *Mobile computing,* pp. 153-181, 1996.

[123] T. Volkert, „Plugin "fog.routing.hrm" für FoGSiEm - Webpräsenz auf GitHub," 2015. [Online]. Available: https://github.com/ICS-TU-Ilmenau/fog/tree/master/fog.routing.hrm. [Zugriff am 23. März 2015].

[124] E. Gamma, R. Helm, R. Johnson und J. Vlissides, Design Patterns - Elements of Reusable Object-Oriented Software, Prentice Hall, 1994.

[125] J. O'Madadhain, D. Fisher, T. Nelson, S. White und Y.-B. Boey, „JUNG - Java Universal Network/graph Framework," 2010. [Online]. Available: http://jung.sourceforge.net/. [Zugriff am 23. März 2015].

[126] C. E. Leiserson, R. L. Rivest, C. Stein und T. H. Cormen, Introduction to algorithms, MIT press, 2001.

[127] S. J. Leffler, R. S. Fabry, W. N. Joy, P. Lapsley, S. Miller und C. Torek, „An Advanced 4.4BSD Interprocess Communication Tutorial," 1986.

[128] T. Hoßfeld, F. Liers, T. Volkert und R. Schatz, „FoG and Clouds: Optimizing QoE for YouTube," *5thGI/ITG KuVS Fachgespräch - NG Service Delivery Platforms*, 2011.

[129] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley und E. Schooler, „SIP: Session Initiation Protocol," IETF RFC 3261, 2002.

[130] „Homer - Webpräsenz auf GitHub," 2015. [Online]. Available: https://github.com/Homer-Conferencing/Homer-Conferencing. [Zugriff am 23. März 2015].

[131] T. Volkert und A. Mitschele-Thiel, „Hierarchical routing management for improving multimedia transmissions and QoE," *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1-3, 2012.

[132] T. Volkert, F. Liers, M. Becke und H. Adhari, „Requirements-oriented path selection for multipath transmission," *12th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks*, 2012.

[133] F. Liers, T. Volkert und A. Mitschele-Thiel, „Scalable network support for application requirements with forwarding on gates," *11th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks*, 2011.

[134] T. Volkert und F. Liers, „Video transcoding and rerouting in Forwarding on Gates networks," *12th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks*, 2012.

[135] T. Volkert, A. Mitschele-Thiel, M. Becke und E. P. Rathgeb, „Homer Conferencing - a Multimedia Test Bed for Various Experiments and Measurements," *7th International Conference on Computing and Convergence Technology (ICCCT)*, pp. 224-229, 2012.

[136] F. Liers, T. Volkert, D. Martin, H. Backhaus, H. Wippel, E. Veith, A. A. Siddiqui und R. Khondoker, „GAPI: A G-Lab Application-to-Network Interface," *11th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks*, 2011.

[137] International Telecommunications Union, „Packet-based multimedia communications," ITU-T H.323, 2009.

[138] Nokia Research Center, „Sofia-SIP Library," 2011. [Online]. Available: http://sofia-sip.sourceforge.net/. [Zugriff am 23. März 2015].

[139] G. Mulligan und D. Gracanin, „A comparison of SOAP and REST implementations of a service based interaction independence middleware framework," *Winter Simulation Conference*, pp. 1423-1432, 2009.

[140] M. Handley, V. Jacobson und C. Perkins, „SDP: Session Description Protocol," IETF RFC 4566, 2006.

[141] H. Schulzrinne, S. Casner, R. Frederick und V. Jacobson, „RTP: A Transport Protocol for Real-Time Applications," IETF RFC 3550, 2003.

[142] F. Fluckiger, Understanding networked multimedia: applications and technologies, Prentice-Hall, 1995.

[143] „German Lab - Webpräsenz," 2008. [Online]. Available: http://www.german-lab.de/. [Zugriff am 23. März 2015].

[144] K. Henke und H.-D. Wuttke, Schaltsysteme: Eine automatenorientierte Einführung, Pearson Studium, 2002.

[145] T. Zinner, T. Hoßfeld, M. Fiedler, F. Liers, T. Volkert, R. Khondoker und R. Schatz, „Requirement driven prospects for realizing user-centric network orchestration," *Multimedia Tools and Applications,* Bd. 74, Nr. 2, pp. 413-437, 2014.

[146] T. Volkert, T. Simon, K. Henke und S. Ostendorff, „IT Infrastructure for Research and Teaching Combining an Online Lab, a UAV and Audio-Visual Communication," *International Journal of Online Engineering,* Bd. 9, Nr. 5, 2013.

[147] M. Faloutsos, P. Faloutsos und C. Faloutsos, „On power-law relationships of the Internet topology," *ACM SIGCOMM Computer Communication Review,* Bd. 29, Nr. 4, pp. 251-262, 1999.

[148] IEEE 802.1 Working Group, „IEEE 802.1D – Spanning Tree Protocol," [Online]. Available: http://standards.ieee.org/getieee802/download/802.1D-2004.pdf. [Zugriff am 23 März 2015].

[149] IEEE 802.1 Working Group, „802.1aq - Shortest Path Bridging," 2012. [Online]. Available: https://standards.ieee.org/findstds/standard/802.1aq-2012.html. [Zugriff am 23 März 2015].

[150] J. Moy, P. Pillay-Esnault und A. Lindem, „Graceful OSPF Restart," IETF RFC 3623, 2003.

[151] L. Nguyen, A. Roy und A. Zinin, „OSPF Restart Signaling," IETF RFC 4812, 2007.

[152] S. Sangli, E. Chen, R. Fernando, J. Scudder und Y. Rekhter, „Graceful Restart Mechanism for BGP," IETF RFC 4724, 2007.

[153] R. Arends, R. Austein, M. Larson, D. Massey und S. Rose, „Resource Records for the DNS Security Extensions," IETF RFC 4034, 2005.

[154] R. Arends, R. Austein, M. Larson, D. Massey und S. Rose, „Protocol Modifications for the DNS Security Extensions," IETF RFC 4035, 2005.

[155] M. Gupta und N. Melam, „Authentication/Confidentiality for OSPFv3," IETF RFC 4552, 2006.

[156] Cisco Systems, „Configuring BGP,“ 2008. [Online]. Available: http://www.cisco.com/c/en/us/td/docs/ios/12_2/ip/configuration/guide/fipr_c/1cfbgp.html. [Zugriff am 23. März 2015].

[157] Cisco Systems, Cisco IOS IP Routing: BGP Command Reference, 2011.

[158] F. Evers, Prädiktive Middleware-basierte Mobilitätsunterstützung für multikriterielle Handover (Dissertation), Universitätsverlag Ilmenau, 2011.

[159] M. Pennewiß, „Konzeption und Umsetzung eines Proxys zur QoS-orientierten Datenübertragung am Beispiel am Beispiel SIP-basierter Videokonferenzen (Bachelorarbeit),“ TU Ilmenau, 2012.

Wissenschaftliche Veröffentlichungen

2014

T. Zinner, T. Hoßfeld, M. Fiedler, F. Liers, T. Volkert, R. Khondoker und R. Schatz, „Requirements driven prospects for realizing user-centric network orchestration," *Multimedia Tools and Applications,* Bd. 74, Nr. 2, pp. 413-437, 2014.

2013

F. Liers, T. Volkert und A. Mitschele-Thiel, „The Forwarding on Gates Architecture: Flexible Placement of QoS Functions and States in Inter-Networks," *International Journal On Advances in Internet Technology,* Bd. 6, Nr. 3, pp. 132-144, 2013.

T. Volkert, T. Simon, K. Henke und S. Ostendorff, „IT Infrastructure for Research and Teaching Combining an Online Lab, a UAV and Audio-Visual Communication, " *International Journal of Online Engineering (iJOE),* pp. 39-44, 2013.

T. Volkert, M. Osdoba, M. Becke und A. Mitschele-Thiel, „Multipath Video Streaming Based on Hierarchical Routing Management," *3rd International Workshop on Protocols and Applications with Multi-Homing Support (PAMS) - 27th International Conference on Advanced Information Networking and Applications (AINA),* pp. 1-6, 2013.

2012

T. Volkert, A. Mitschele-Thiel, M. Becke und E. P. Rathgeb, „Homer Conferencing - a Multimedia Test Bed for Various Experiments and Measurements," *7th International Conference on Computing and Convergence Technology (ICCCT),* pp. 224-229, 2012.

T. Volkert, F. Liers, M. Becke und H. Adhari, „Requirements-oriented path selection for multipath transmission," *12th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks,* 2012.

T. Volkert und F. Liers, „Video transcoding and rerouting in Forwarding on Gates networks," *12th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks,* 2012.

F. Liers, T. Volkert und A. Mitschele-Thiel, „The forwarding on gates architecture: Merging intserv and diffserv," *4th International Conference on Advances in Future Internet (AFIN),* pp. 7-13, 2012.

T. Hoßfeld, F. Liers, R. Schatz, B. Staehle, D. Staehle, T. Volkert und F. Wamser, „Quality of Experience Management for YouTube: Clouds, FoG and the AquareYoum," *PIK - Praxis der Informationsverarbeitung und Kommunikation,* Bd. 35, Nr. 3, pp. 133-143, 2012.

T. Volkert und A. Mitschele-Thiel, „Hierarchical routing management for improving multimedia transmissions and QoE," *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM),* pp. 1-3, 2012.

T. Zinner, T. Hoßfeld, D. Rauscher, B. Reuther, D. Günther, T. Volkert, F. Liers und M. Fiedler, „Prospects for Realizing User-Centric Network Orchestration: FEC-protected SVC Streaming," *7th KuVS Workshop on Future Internet,* 2012.

2011

T. Hoßfeld, F. Liers, T. Volkert und R. Schatz, „FoG and Clouds: Optimizing QoE for YouTube," *5thGI/ITG KuVS Fachgespräch - NG Service Delivery Platforms,* 2011.

F. Liers, T. Volkert und A. Mitschele-Thiel, „Scalable network support for application requirements with forwarding on gates," *11th Würzburg Workshop on IP: ITG Workshop - Visions of Future Generation Networks,* 2011.

F. Liers, T. Volkert, D. Martin, H. Backhaus, H. Wippel, E. Veith, A. A. Siddiqui und R. Khondoker, „GAPI: A G-Lab Application-to-Network Interface," *11th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks,* 2011.

2010

F. Liers, T. Volkert und A. Mitschele-Thiel, „On Routing with Forwarding on Gates," *10th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks,* 2010.

K. Henke, S. Ostendorff, T. Volkert und A. Mitschele-Thiel, „A Universal Communication Framework and Navigation Control Software for Mobile Prototyping Platforms," *International Journal of Online Engineering (iJOE),* Bd. 6, pp. 19-24, 2010.

F. Liers, T. Volkert und A. Mitschele-Thiel, „Demonstrating Forwarding on Gates with First Applications," *10th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks,* 2010.

K. Henke, S. Ostendorff, T. Volkert und A. Mitschele-Thiel, „A Universal Communication Framework and Navigation Control Software for Mobile Prototyping Platforms," *REV2010-Remote Engineering & Virtual Instrumentation,* 2010.

M. Hein, A. Kraus, R. Stephan, C. Volmer, A. Heuberger, E. Eberlein, C. Keip, M. Mehnert, A. Mitschele-Thiel, P. Drieß und T. Volkert, „Perspectives for Mobile Satellite Communications in Ka-Band (MoSaKa)," *EuCAP'2010: The 4th European Conference on Antennas and Propagation,* 2010.

2009

K. Henke, S. Ostendorff, T. Volkert und A. Mitschele-Thiel, „Flying WiFi Robots as Mobile Platforms," *Computers and Advanced Technology in Education (CATE 2009),* pp. 14-21, 2009.

K. Henke, S. Ostendorff, T. Volkert und A. Mitschele-Thiel, „Mobile Prototyping Platforms for Remote Engineering Applications," *International Journal of Online Engineering (iJOE),* Bd. 5, pp. 35-42, 2009.

F. Liers, T. Volkert und A. Mitschele-Thiel, „Forwarding on Gates: A clean-slate Future Internet Approach within the G-Lab project," *9th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks,* 2009.

K. Henke, S. Ostendorff, T. Volkert und A. Mitschele-Thiel, „Mobile Prototyping Platforms for Remote Engineering Applications," *6th International Conference on Remote Engineering and Virtual Instrumentation,* 2009.

M. A. Kalil, F. Liers, T. Volkert und A. Mitschele-Thiel, „A Novel Opportunistic Spectrum Sharing Scheme for Cognitive Ad Hoc Networks, " *5th Workshop on Mobile Ad-Hoc Networks (WMAN) – in*

conjunction with the 16th bi-annual Conference on Communication in Distributed Systems (KiVS), 2009.

2008

F. Liers, T. Volkert und A. Mitschele-Thiel, „A Flexible Abstraction for the Future Internet," *8th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop - Visions of Future Generation Networks,* 2008.

F. Liers, T. Volkert und A. Mitschele-Thiel, „A New Forwarding Approach for the Future Internet," *1.ICSY Fachgespräch der GI/ITG-Fachgruppe „Kommunikation und Verteilte Systeme",* 2008.

Betreute studentische Arbeiten

[1] Manuel Osdoba: **Evaluierung eines hierarchischen Routingsystems im Kontext des Future Internet Ansatzes „Forwarding on Gates“** (*Masterarbeit*)

[2] Stefan-Wieland Kögel: **Hierarchisches Routing im Kontext des Future Internet Ansatzes "Forwarding on Gates"** (*Diplomarbeit*)

[3] Alexander Krause: **Evaluierung der Netzwerkprotokollsuite im Linuxkernel mit Schwerpunkt auf Leistungsfähigkeit und deren Beeinflussung** (*Diplomarbeit*)

[4] Frank Roth: **Interoperabilität zwischen dem Internet Protokoll IP und dem Future Internet Ansatz "Forwarding on Gates"** (*Diplomarbeit*)

[5] Thomas Hertwig: **Konzept und Realisierung einer zentralen Regeleinheit für mobile Fluggeräte auf FPGA** (*Diplomarbeit*)

[6] Martin Werner: **Modellierung und Analyse eines Provider-basierten Systems zur Bereitstellung und Verwaltung von sehr großen WLAN-Netzen** (*Diplomarbeit*)

[7] Marcel Pennewiß: **Konzeption und Umsetzung eines Proxys zur QoS-orientierten Datenübertragung am Beispiel SIP-basierter Videokonferenzen** (*Bachelorarbeit*)

[8] Thomas Dietrich: **Leistungsbemessung und -bewertung von Protokollen auf Nutzerbene** (*Bachelorarbeit*)

[9] Steffen Werfel: **Vergleich bekannter "Future Internet"-Ansätze anhand vorhandener Implementierungen** (*Bachelorarbeit*)

[10] Stefan-Wieland Kögel: **Evaluierung von QoS-Managementarchitekturen für heterogene Satelliten-basierte Netzwerke** (*Studienarbeit*)

[11] Sandro Pax: **Konfiguration und Monitoring mobiler Kommunikationsplatformen** (*Studienarbeit*)

[12] Frank Roth: **Vergleich verfügbarer Applikationen zur Analyse von Kommunikationsströmen in heutigen Netzwerken** (*Studienarbeit*)

[13] René Hutschenreuter: **Kompaktes Routing in heutigen und zukünftigen Netzwerken** (*Hauptseminar*)

[14] Sven Biegler: **Grafische Positionsüberwachung einer mobilen Kommunikationsplatform** (*Hauptseminar*)

[15] Markus Filzhuth: **Einsatzmöglichkeiten mobiler Kommunikationsplatformen für Remoteengineering-Anwendungen** (*Hauptseminar*)

[16] René Hutschenreuter: **QoS-Management auf hierarchischem Routing** (*Hauptseminar*)

[17] Johannes Both: **Source Routing - Allgemeine Routingansätze** (*Hauptseminar*)

[18] Florian Frisch: **Source Routing - Angriffsmöglichkeiten** (*Hauptseminar*)

[19] Folker Schwesinger: **Source Routing - Erweiterungen gegen Angriffsmöglichkeiten** (*Hauptseminar*)

[20] Christian Heinz: **Source Routing - Optimierungsansätze** (*Hauptseminar*)

[21] Marcel Pennewiß: **Übertragungsparameter und Dienstgüte (QoS) im Kontext von SIP-basierenden Videokonferenzsystemen** (*Hauptseminar*)

[22] Jacqueline White: **Vergleich bekannter Ansätze für inkrementelles Routing** (*Hauptseminar*)

[23] Lei Bai, Wenjie Cai, Shuang Zhang: **Verteilung von Routing-Informationen in heutigen Netzwerken** (*Hauptseminar*)

[24] Freddy Herzog, David Kling: **Erweiterung der Sensorik einer mobilen Kommunikationsplattform** (*Projektseminar*)

[25] Kay Wenzel: **Möglichkeiten und Grenzen der GPS-Autonavigation** (*Projektseminar*)

[26] Sven Biegler: **Remote-Monitoring einer mobilen Kommunikationsplatform** (*Projektseminar*)

[27] Thomas Dietrich: **Untersuchung alternativer Kommunikationsschnittstellen einer mobilen Kommunikationsplatform** (*Projektseminar*)

[28] Raik Schulze: **Untersuchung der Steueralgorithmen einer mobilen Kommunikationsplattform** (*Projektseminar*)

[29] Markus Hartwig, Michael Kirchhoff: **Paketinjektion und -empfang innerhalb einer Java-Umgebung** (*Projektseminar*)

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt.

Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch bewertet wird und gemäß §7 Abs. 10 der Promotionsordnung den Abbruch des Promotionsverfahrens zur Folge hat.

Ilmenau, den 26.Juni 2015

(Thomas Volkert)